Real Time Human Tracking in Unconstrained Environments

A thesis

submitted in partial fulfilment of

the requirements for the Degree

of

Doctor of Philosophy

in the

University of Canterbury

by

Hongzhi Gao

University of Canterbury

2011

Publications

Gao, H. and Green, R., *Application of Tabu Search Optimization in Real-time Video Tracking.* In Proceedings of the 24th International Conference on Image and Vision Computing New Zealand. Wellington, New Zealand. November 2009

Gao, H. and Green, R., *A Sequential Monte Carlo Method for Particle Filters.* In Proceedings of the 23rd International Conference on Image and Vision Computing New Zealand. Christchurch, New Zealand. November 2008

Gao, H. and Green, R., *A Robust Moving Object Segmentation Algorithm.* International Conference on Wavelet Analysis and Pattern Recognition, 2007. (ICWAPR '07). Volume 1, 2-4 Nov. 2007 Page(s):214 – 217.

Gao, H. and Green, R., *A Quantitative Comparison Research on Frame Level Background Subtraction Algorithms.* In Proceedings of the 22nd International Conference on Image and Vision Computing New Zealand. Hamilton, New Zealand, November, 2007

Gao, H. and Lim, M. and Lin, E. and Green, R., *Marker based Facial Tracking Application in Communication Disorder Research.* In Proceedings of the 22nd International Conference on Image and Vision Computing New Zealand. Hamilton, New Zealand, November, 2007

Gao, H. and Green, R., *Real-Time Predictive 3D Tracking of Cricket Balls For Knowledge of Results and Strategy Training.* 8th Australasian Conference on Mathematics and Computers in Sport, Australia, July, 2006.

Gao, H. and Green, R., *A Robust Efficient Motion Segmentation Algorithm.* In Proceedings of the 21st International Conference on Image and Vision Computing New Zealand. Auckland, New Zealand, November, 2006

Abstract

The tabu search particle filter is proposed in this research based on the integration of the modified tabu search metaheuristic optimization and the genetic particle filter. Experiments with this algorithm in real time human tracking applications in unconstrained environments show that it is more robust, accurate and faster than a number of other existing metaheuristic filters, including the evolution particle filter, particle swarm filter, simulated annealing filter, path relink filter and scatter search filter. Quantitative evaluation illustrates that even with only ten particles in the system, the proposed tabu search particle filter has a success rate of 93.85% whereas the success rate of other metaheuristic filters ranged from 68.46% to 17.69% under the same conditions. The accuracy of the proposed algorithm (with ten particles in the tracking system) is 2.69 pixels on average, which is over 3.85 times better than the second best metaheuristic filters in accuracy and 18.13 times better than the average accuracy of all other filters. The proposed algorithm is also the fastest among all metaheuristic filters that have been tested. It achieves approximately 50 frames per second, which is 1.5 times faster than the second fastest algorithm and nineteen times faster than the average speed of all other metaheuristic filters.

Furthermore, a unique colour sequence model is developed in this research based on a degenerated form of the hidden Markov model. Quantitative evaluations based on rigid object matching experiments illustrate that the successful matching rate is 5.73 times

better than the widely used colour histogram. In terms of speed, the proposed algorithm achieves twice the successful matching rate in about three quarters of the processing time consumed by the colour histogram model.

Overall, these results suggest that the two proposed algorithms would be useful in many applications due to their efficiently, accuracy and ability to robustly track people and coloured objects.

List of Abbreviations

- **HSV**: Hue Saturation Value
- **RGB**: Red Green Blue
- **ROI**: Region of Interest
- **AR**: Augmented Reality
- **EKF**: Extended Kalman Filter
- **UKF**: Unscented Kalman Filter
- **CSM**: Colour Sequence Model
- **CHMR**: Continuous Human Movement Recognition
- **HMM**: Hidden Markov Model
- **CHM**: Colour Histogram Model
- **CMR**: Correct Matching Rate
- **EV**: Evolutionary
- **SA**: Simulated Annealing
- **PS**: Particle Swarm
- **SS**: Scatter Search
- **PR**: Path Relink

Table of Contents

Table of Figures

Chapter 1
Introduction

The scope of this research is to improve the accuracy of the particle filter [1] with minimum performance penalty.

In this research, the term:"unconstrained environment" refers to urban outdoor environments in uncontrolled weather conditions. The targets of interest are random pedestrians on the street at the time when testing footages were captured. Only pixel-wise colours are considered as target and background features.

Limitations of real time tracking in unconstrained environment are:

1. High efficient tracking algorithms are required to achieve real time performance.
2. Assumption that the global optimum of the observation model is the target of interest may be incorrect because of the cluttered background.
3. No assumption shall be made in terms of the colour of the tracking targets.

In this research, the following two contributions are made to overcome the shortcomings identified in the computer vision based tracking system proposed in the past:

1. Developing a new metaheuristic particle filter that outperforms existing algorithms in this category in reliability, accuracy and speed.
2. Creation of a unique colour sequence model for handling the illumination invariance problem in red-green-blue colour space. The proposed model outperforms the colour histogram model in both accuracy and speed.

In the context of computer vision based object tracking system, improvement in the particle filter can be classified into three directions.

Firstly, variations of the stochastic predictive process have been used to improve the tracking accuracy and/or tracking performance of the particle filter. For example the Rao-Blackwellised particle filter [4] addressed the high computation cost usually found in high dimensional state space tracking systems [4, 5] by decomposing the state space of the system.

Furthermore, a set of combined filters have been found to improve the accuracy of the particle filter. For example, the Kalman particle filter [6], the extended Kalman particle filter [7], the iterative extended Kalman particle filter [8], the unscented particle filter [9], the iterative unscented particle filter [10] and the divided difference particle filter [11].

Recently, a new research trend has emerged in this area, which is to improve the quality of the particle filter by integrating metaheuristic optimization. Algorithms already

being proposed in the literature in the category include: the annealing particle filter [12], the particle swarm particle filter [13], the evolution particle filter [14, 15], the scatter search particle filter and the path relink particle filter [16].

Despite some positive results, there is no prior research that adequately explained how to choose an adequate metaheuristic optimisation algorithm to combine with the particle filter. In early research, metaheuristic algorithms were chosen in an ad hoc manner, which means that neither the optimised accuracy nor the optimised speed can be achieved.

In this research, a novel metaheuristic particle filter is proposed. The proposed algorithm achieves better reliability, accuracy and speed by: 1) imposing connections between the objective function of the metaheuristic process and the motion model of the particle filtering process; 2) building data structures to store all visited solutions so as to improve performance by reducing duplicated visits to the same solution.

The colour histogram model is a widely used presentation of the targets in a computer vision based tracking system. The colour histogram model does not have illumination invariant capability and therefore, an illumination invariant colour space, for example the hue-saturation-value colour space, is usually used in a colour histogram based system to improve the robustness of the tracking system in the environment with variant lighting strength.

However, an implicit assumption of such system is that the target of interest is wearing colours of high saturation – this assumption may be too strict as black, gray and white colours are very common in modern society.

In this research, a novel colour sequence model is proposed. The proposed colour model achieves illumination invariant at the colour model level (as opposite to the colour space level commonly used in the literature) by imposing dependency between adjacent colour regions. Therefore illumination invariant can be achieved even red-green-blue colour space is used in the system.

The rest of this thesis is organized as follows. Firstly, background knowledge in related to the proposed algorithms are introduced in the following three chapters. Each background chapter discusses one specific topic that is relevant to the proposed algorithms but do not have strong connection among themselves. Chapter 2 reviews earlier computer vision based tracking systems. Chapter 3 introduces Bayesian filters that are commonly used in tracking systems. Chapter 4 introduces the metaheuristic optimisation algorithms that have been incorporated with particle filters in the past.

Then, the proposed algorithms are discussed in the two chapters followed by the background chapters. That is: chapter 5 introduces a new metaheuristic particle filter, the tabu search particle filter and chapter 6 presents a unique colour sequence model.

Finally, the experimental results are discussed in chapter 7 and then the entire thesis is concluded in chapter 8.

Chapter 2
Object Tracking

The complexity of tracking algorithm relies on the properties of its application. In the simplest situation, if the target has globally unique visual features and real time performance is not required, then the tracking can be achieved by associating target's detection results in a sequence of images. However, if the target cannot be reliably detected from the entire image and/or the time of processing the entire image is not acceptable, then the temporal history of the target (trajectory) must be taken into consideration to minimise tracking errors and improve tracking speed by pruning the search space of the target.

In this chapter, object tracking systems are discussed in ascending order of complexity. Firstly, in section 2.1, simple tracking applications for globally unique targets are introduced. In section 2.2, the tracking algorithms that take noisy image segmentation results as input are described. Finally, some previous research of tracking in unconstrained environments are discussed in section 2.3.

2.1 Tracking Globally Unique Targets

If a target has some properties that can be used to uniquely identify its position, then target tracking can be achieved by associating feature detection results along a sequence of images.

An example of such an application is illustrated in figure 2.1 [18]. The research shown here is a human computer interface using hand gestures. As shown, in figure 2.1 (third image on the top row), the tracking area is covered by black non-reflective fabric and the human operator is also wearing black clothing. In such a strictly controlled environment, the skin colour of the operator's right hand becomes a unique feature inside the region covered by the black non-reflective fabric, which can be used to identify its position. The rest five images in figure 2.1 illustrate other processing steps being carried in this system; including: individual image analysis steps, which have to deal with imperfect segmentation and correct possible errors at each successive stage of feature finding, point matching, finger direction finding, and collision detection. These steps are not relevant to the thesis goal and therefore not being discussed in this chapter.

The main limitation of this tracking algorithm is that a strictly controlled environment is required so that the uniqueness of the target is maintained throughout the whole tracking period.

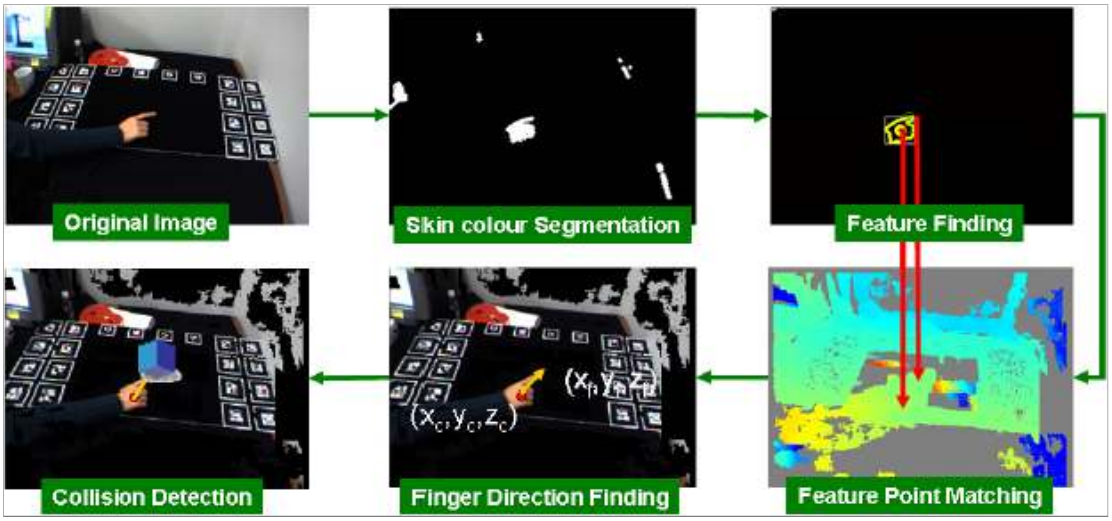


Figure 2.1 [18]: A colour based hand tracking system for human computer interaction research.

2.2 Tracking Object Being Represented by Point

In some applications, the target is represented by a point [101] that is the centroid of the occupied region acquired by image segmentation algorithms.

The general architecture of these algorithms is depicted in figure 2.2 [20]. As shown, images are initially processed by a "detector", which segments an image into background and foreground regions based on predefined classification conditions, such as motion.

The segmented foreground areas, which include the real target and some incorrectly segmented regions from the background, are presented as points (centroids) and

delivered to the “association” module, so that they can be associated with existing tracks. Usually, not all detected points can be associated to existing tracks because some of them represent noise or other incorrectly segmented objects. On the other hand, not all existing tracks can be associated with a valid foreground point because the target may leave the scene or be completely occluded by some background structure.

In the “initialise/delete” process new tracks are initialised and old tracks are terminated. Finally, the tracks are updated and passed back to the next processing iteration.

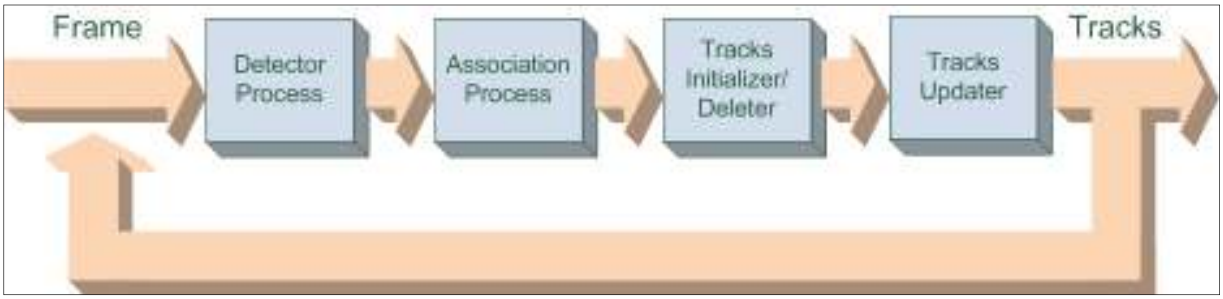


Figure 2.2 [20]: The standard architecture of tracking algorithms that belong to the “tracking with foreground / background segmentation” category.

Figure 2.3 [21] illustrates an example of the association process. In this example, four observations are detected in the validation regions (as the surrounding area of the estimated target position, illustrated with ellipses in figure 2.3) of two known targets Z^1 and Z^2. Both observations $\star Z_1$ and $\star Z_2$ may or may not be associated with the target Z^1. Similarly, one observation among $\star Z_2$, $\star Z_3$ and $\star Z_4$ can be associated with

the target Z^2, or none of them may be Z^2. Furthermore, the observation $\star Z_2$ can be associated to both tracks when the targets overlap.

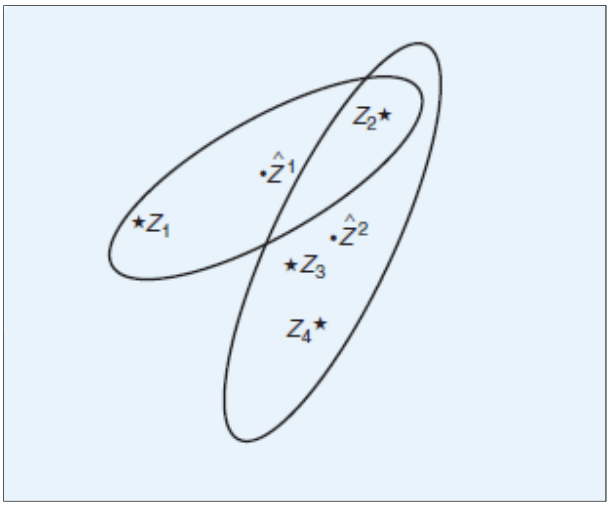


Figure 2.3 [21]: An example of the data association dilemma in multiple target tracking.

Some algorithms have been developed to optimize the solution of this data association problem, such as the nearest neighbour rule [22] or the multiple hypothesis tracking algorithm [23]. Another solution is the joint probability data association filter [24], which combines the probability based soft data association within a particle filter in multiple target tracking.

Figure 2.4 illustrates an multiple object tracking system that tracks humans and vehicles simultaneously using multiple cameras in outdoor environment [99]. In this research, Foreground segmentation is performed using the codebook method [28] and HSV

colour space is used. The proposed algorithm in [99] achieves 67.9% of successful tracking rate at approximately 10 frames in a second.

Figure 2.4: Multiple objects tracking system [99].

The main limitation of the tracking algorithms discussed is that the quality of tracking processes is that these algorithms heavily depend on the quality of the image segmentation algorithm. Consequently, these tracking algorithms cannot be used in applications where good image segmentation is difficult to achieve, such as video captured with moving camera.

2.3 Tracking with Filtering Algorithms

The accuracy of the tracking system may decrease dramatically in unconstrained environments if only the image segmentation based tracking algorithms discussed in the

previous section are used. This is because most image segmentation algorithms are not robust enough to maintain sufficient quality over time in unconstrained environments. Filtering based algorithms, for example the particle filter that will be discussed in detail in chapter 3, are designed to improve the tracking accuracy in such situation because the previous knowledge of the target's position and the properties of its movement are used to assist the feature matching (e.g. colour comparing) process.

A range of filtering based tracking algorithms has been developed in earlier research to tackle the difficulty of tracking in outdoor public spaces [101, 102]. In these algorithms, the trajectory of the target is used to predict the target's position in the next image. The estimated target state (which may include position and other information) is represented by a probability distribution. After the estimation step, the correctness of the predicted target state is measured using a feature comparison algorithm, such as colour histogram, when the new image is received.

Figure 2.5 illustrates a full body human tracking system developed in [35]. In this research, a dynamic kernel-based progressive particle filter is proposed. The proposed algorithm decomposes the high dimensional parameter space into several low dimensional ones [35]. Moreover, the proposed algorithm applies multiple predications and a mean shift tracker with dynamic allocated band width to estimate the human posture iteratively [35]. The tracking system proposed in this research is designed for

strictly controlled environment (for example a dedicated computer vision lab) and it is not running in real time.

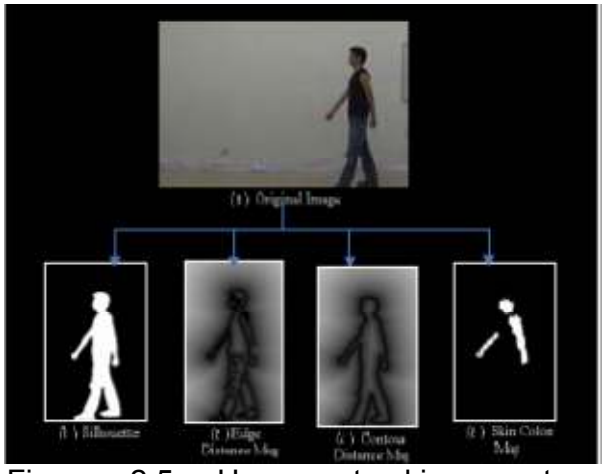


Figure 2.5: Human tracking system proposed in [35].

Figure 2.6 illustrates the tracking result of an indoor human tracking system developed in [98]. In this research, the face areas of human targets are extracted based on a skin colour based model [98]. Then the face position in the video is tracked by a particle filter based visual tracker and both the position and the velocity of the targets are obtained from the tracker [98]. The proposed algorithm in [98] is developed for indoor environment only and it requires manual extraction of the facial region.

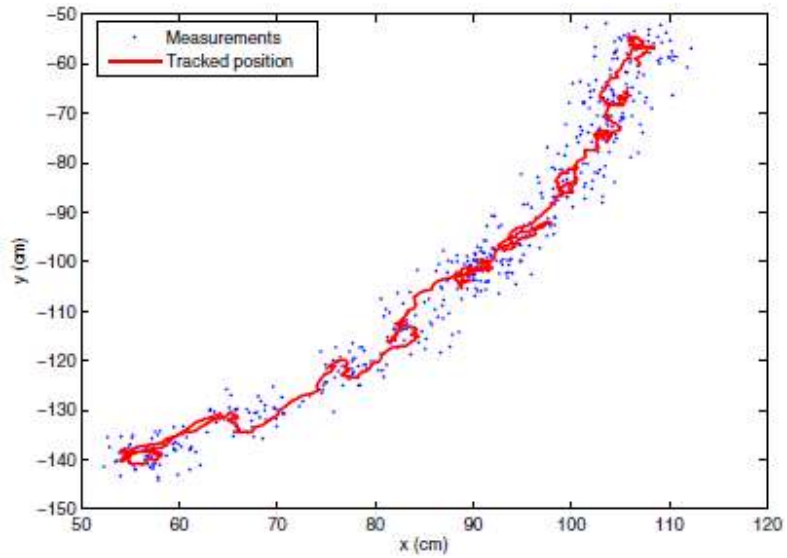


Figure 2.6: Experimental results of the human tracking system proposed in [98]

Figure 2.7 illustrates a particle filter based human tracking application developed in [31]. In this system, targets are tracked in a two-step process. Firstly, the position of a target is estimated based on the knowledge of its previous position and a pre-defined motion model. Because the particle filter is used in this application, the estimated positions of the target in the coming image are represented by a set of 150 Monte Carlo samples (particles). Secondly, the correctness of these particles, represented by their weights, is measured by the Bhattacharyya distance between the target's colour histogram and the particle's colour histogram. The position of this target in the new image is calculated from the most accurate estimate, namely the particle with the highest weight.

Figure 2.7 [31]: Example of a particle filter based human tracking system.

Figure 2.8 illustrates another particle filter based human tracker from [32]. In this research, not only the target's motion and appearance properties are taken into consideration during the tracking process but also information about the structured environment. As shown, the three images in the bottom row illustrate the likelihood of a target appearing, where targets are more likely to be found in the blue region than the red region.

Experimental results from [32] indicate that their algorithm is more accurate than the generic particle filter [31]. However, the speed of this algorithm is five times slower than the generic particle filter for the same number of particles.

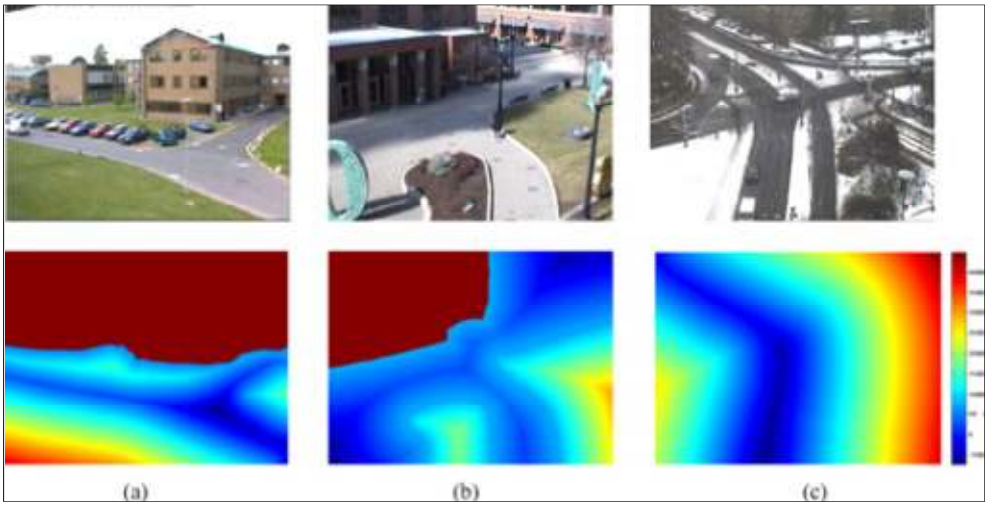


Figure 2.8 [32]: Object tracking in structured environments with a particle filter with the bottom row indicating position likelihood from red (unlikely) to blue (likely).

The main limitation of the particle filter based tracking system is that a large number of particles are required to be used to ensure sufficient accuracy, but culminating in a significant performance penalty. For example, the algorithm from [32] needs 150 particles to track one target and it achieves only 2.5 frames per second

2.4 Summary

Commonly used object tracking algorithms in the computer vision research are introduced in this chapter. For the simplest case where the target has globally unique visual features across the whole sequence of images, the tracking algorithm can be designed to use target detection results across all images. However, these algorithms

can only be applied in strictly controlled environments. Furthermore, if the segmentation algorithm is applied in a system to generate good quality foreground image, then the tracking algorithm can be based on associating the previously known trajectory with the new detected foreground object. However, the tracking may fail if the image segmentation results are too noisy or the target is invisible due to object occlusion. Finally, tracking algorithms based on stochastic filters has been used in complex environments where image segmentation is difficult to achieve. The accuracy and performance of these algorithms depend on the quality of the stochastic filter being used.

Chapter 3
Bayesian based Tracking

The Bayesian filter series are widely used in computer vision based tracking systems. This chapter introduces such predictive tracking algorithms including the Bayesian filter, three members of the Kalman filter family and the particle filter.

3.1 Bayesian Filter

An object tracking problem can be solved by a two-step recursive algorithm [34] with an estimation step and a measurement step. In the estimation step, the prior belief of a target's state vector, such as an estimation of an object's position in the coming frame, is calculated based on some known state vectors of this object in the past. In the measurement step, the posterior belief of the target's state vector is calculated by updating the prior belief with the latest observation. Equations (3.1) and (3.2) specify a stochastic process of transitions between the unobserved noisy states, represented by observed noisy measurements.

The estimation of the state vector in a tracking system can be defined as a recursive discrete stochastic process, and be presented as:

$$x_t = f_{t-1}(x_{t-1}, v_{t-1}) \quad (3.1)$$

In this equation (3.1), $f_{t-1}(\cdot)$ defines prior knowledge of the target's dynamic behaviours, which is usually called the motion model. The unknown and unpredictable behaviours of the target, which cannot be described accurately by a motion model, are represented by the process noise factor v_{t-1}. The state of the target is represented as a vector and nominated as $x_t \in R^n$, where n is the dimension of the state vector and t is the time index.

The measurement process of a tracking system can be presented as:

$$z_t = h_t(x_t, w_t) \quad (3.2)$$

In this equation (3.2), $h_t(\cdot)$ defines prior known rules of mapping observable quantities in the tracking system to the state vector of the target. The unknown or unpredictable interference of the observation process are represented by the measurement noise w_t. The measurement of the target is represented as a vector and nominated as $z_t \in R^m$, where m is the dimension of the measurement vector.

The Bayesian estimate of an unobserved state at time t depends generally on all the available measurements up to this time, but only due to an assumed Markov chain model of state transitions in equation 3.1 and independent measurements of equation 3.2 it can be found recursively. In a Bayesian filtering system, the two step recursive algorithm introduced above is presented as stochastic processes.

The estimation step can be presented as a stochastic process in equation 3.3 [34]:

$$\overline{bel(x_t)} = p(x_t|z_{t-1}) = \int [p(x_t|x_{t-1})\, p(x_{t-1}|z_{t-1})]\, dx_{t-1} \tag{3.3}$$

In this equation, the symbol $\overline{bel(x_t)}$ indicates the prior belief of the target's state vector x_t. The state transition probability distribution $p(x_t|x_{t-1})$ specifies how state vector evolves over time, which is given by $f_t(\cdot)$ in equation 3.1. The previous posterior belief of the state vector, namely $bel(x_{t-1}) = p(x_{t-1}|z_{t-1})$, in the second clause of this equation is considered to be known because of the recursive nature of this process.

The measurement step is presented as a stochastic process in equation 3.4:

$$\begin{aligned} bel(x_t) &= \eta\, p(z_t|x_t)\, p(x_t|z_{t-1}) \\ &= \eta\, p(z_t|x_t)\, \overline{bel(x_t)} \end{aligned} \tag{3.4}$$

In this equation, the symbol $bel(x_t)$ indicates the posterior belief of the target's state vector, which is given by a combined factor of the prior belief $\overline{bel(x_t)}$ and the measurement probability distribution $p(z_t|x_t)$. The measurement probability distribution specifies how the state vector can be verified through some observable quantities, which are given by $h_t(\cdot)$ in equation 3.2. The normalising factor η in equation 3.4 converting a conditional belief into an unconditional belief for the known measurements.

In a Bayesian filtering system, given the posterior belief of a state vector, the optimal state vector of the target can be calculated by two methods: the minimum mean

squared error (MMSE) algorithm and the maximum a posteriori (MAP) algorithm. The MMSE algorithm defines the optimised state vector of the target as the weighted mean of its posterior belief, which can be represented as equation 3.5 [34]:

$$\widehat{x_{t|t}}^{MMSE} \triangleq E\{x_t|z_t\} = \int [x_t\, p(x_t|z_t)]\, dx_t \tag{3.5}$$

The MAP algorithm defines the optimised state vector of the target as the maximum weighted posterior belief, which can be represented as equation 3.6:

$$\widehat{x_{t|t}}^{MAP} \triangleq \underset{x_t}{\operatorname{argmax}}\, p(x_t|z_t) \tag{3.6}$$

3.2 Kalman Filters

The Kalman filter family [33-40] offers optimal solutions for a subset of problems whose belief distributions are assumed to be Gaussian. Some of the commonly used filters belong to this family are introduced in this section, which includes the Kalman filter [34], the extend Kalman filter (EKF) [41] and the unscented Kalman filter (UKF) [39].

3.2.1 The Kalman Filter

Problems to be optimally solved by the Kalman filter have two constraints. Firstly, the belief distributions of state vector are assumed to be Gaussian. Secondly, the transition functions in both the estimation process and the measurement process are assumed to be linear.

The first assumption, namely the Gaussian constraint [35], indicates that the state vector x_t in the system shall be determined by a mean vector μ_t and a covariance matrix Σ_t, where the state vector x_t can be described mathematically as equation 3.7:

$$f(x_t\,;\,\mu_t,\,\Sigma_t)\;=\;\frac{exp\left\{-\frac{1}{2}\,(x_t-\,\mu_t)^T\;{\Sigma_t}^{-1}\,(x_t-\mu_t)\right\}}{det(2\pi\,\Sigma_t)^{\frac{1}{2}}} \tag{3.7}$$

For example, in the context of an object tracking system, the Gaussian constraint of a two dimensional state vector $(x,y)^T$ can be visualized as a single position annotated with an uncertainty ellipse.

The second assumption, namely the linear constraint [35], indicates that the transition functions defined in equation 3.1 and 3.2 can be presented in linear forms.

The state estimation function defined in equation 3.1 is redefined in its linear form as equation 3.8:

$$x_t\;=\;A_t x_{t-1}+\;\varepsilon_t \tag{3.8}$$

In this equation, A_t is a square matrix of size n in both dimensions, which encodes the motion model $f_t(\cdot)$ of the system, and ε_t is a zero mean Gaussian random vector, which encodes the process noise v_t. The covariance matrix of this noise vector ε_t is denoted as R_t and is referenced in equation 3.10.

The measurement function defined in equation 3.2 is redefined in its linear form as equation 3.9:

$$z_t = C_t x_t + \delta_t \tag{3.9}$$

In this equation, C_t is a matrix of size $m\ by\ n$, which encodes the measurement model $h_t(\cdot)$ of the system, and δ_t is a zero mean Gaussian random vector, which encodes the observation noise w_t. The covariance matrix of the observation noise vector δ_t is denoted as Q_t and is referenced in equation 3.11.

Following both assumptions, the estimation process in the Bayesian filter defined in equation 3.3 can be reinterpreted in its linear Gaussian form as equation 3.10:

$$p(x_t|x_{t-1}) = \frac{exp\left\{-\frac{1}{2}\ (x_t - A_t x_{t-1})^T\ \mathrm{R_t}^{-1}\ (x_t - A_t x_{t-1})\right\}}{det(2\pi \mathrm{R_t})^{\frac{1}{2}}} \tag{3.10}$$

Similarly, the measurement process in the Bayesian filter defined in equation 3.4 can also be reinterpreted in its linear Gaussian form as equation 3.11:

$$p(x_t|z_t) = \frac{exp\left\{-\frac{1}{2}\ (z_t - C_t x_t)^T\ \mathrm{Q_t}^{-1}\ (z_t - C_t x_t)\right\}}{det(2\pi Q_t)^{\frac{1}{2}}} \tag{3.11}$$

Both the estimation process and the measurement process can be calculated in closed form with their linear Gaussian definitions.

A single iteration of the Kalman filter is illustrated in figure 3.1. As shown, the Kalman process takes two input parameters. Firstly, the posterior belief distribution of the

previous time step $bel(x_{t-1})$ defined in its Gaussian form by the mean vector μ_{t-1} and the covariance matrix Σ_{t-1}. Secondly, the observation vector of current time step z_t. The return value of the Kalman filter is the posterior belief distribution at the current time step $bel(x_t)$ defined in its linear Gaussian form by the mean vector μ_t and the covariance matrix Σ_t.

Ordinary Kalman Filter:

Input parameters: μ_{t-1}, Σ_{t-1}, z_t; *Return values:* μ_t, Σ_t

BEGIN

$$\overline{\mu_t} = A_t \mu_{t-1}$$

$$\overline{\Sigma_t} = A_t \Sigma_{t-1} A_t^T + R_t$$

$$K_t = \overline{\Sigma_t} C_t^T \left(C_t \overline{\Sigma_t} C_t^T + Q_t \right)^{-1}$$

$$\mu_t = \overline{\mu_t} + K_t (z_t - C_t \overline{\mu_t})$$

$$\Sigma_t = (I - K_t C_t) \overline{\Sigma_t}$$

RETURN μ_t, Σ_t

END

Figure 3.1: The pseudo code of Kalman filter [37].

As shown in figure 3.1, the Kalman filter is divided into three colour coded sections. They are the prior belief estimation process (red colour section), the Kalman gain calculation process (green colour section) and the posterior belief calculation process (blue section).

In the prior belief estimation process, the prior belief distribution $\overline{bel(x_t)}$ defined in its linear Gaussian form by $\overline{\mu_t}$ *and* $\overline{\Sigma_t}$ are estimated based on the known posterior belief of

the previous time step $bel(x_{t-1})$ and the knowledge of the motion model. As illustrated in the red coloured section in figure 3.1, the mean vector $\overline{\mu_t}$ is calculated through a noise free linear process with predefined motion model and the noise of this vector is encoded in the estimated covariance matrix $\overline{\Sigma_t}$.

An intermediate variable, the Kalman gain K_t, is defined in the green colour section in figure 3.1. This variable indicates the degree to which the observation is incorporated into the posterior belief based on the reliability of the measurement process.

As shown in the blue colour section in figure 3.1, the posterior belief of the state vector of the current time step is calculated based on incorporating the prior belief with the current observation in proportion to the Kalman gain.

3.2.2 The Extended Kalman Filter

The accuracy of the Kalman filter relies heavily on the linearity of the system. However, being linear is a strong constraint and is usually unrealistic in an object tracking system,

The Extended Kalman Filter (EKF) improves the Kalman filter by relaxing the linear constraint of the state transition functions and only assuming that both functions, namely $f_t(\cdot)$ in state transition process in equation 3.1 and $h_t(\cdot)$ in the measurement process in equation 3.2, are differentiable [42].

The Gaussian assumption of the state vector remains in the EKF. Similar to the Kalman filter, the state vector x_t in an EKF system is also determined by its mean vector μ_t and covariance matrix Σ_t. In order for the belief distributions to be Gaussian after the state transition process, the nonlinear state transition functions in the EKF are linearlized at a selected point because state transition through a nonlinear function damages the Gaussian property of a distribution. Therefore, the state transition in EKF is an approximation process. Among all linearization algorithms, the first order Taylor expansion algorithm is usually applied in EKF to linearize state transition functions and the actual expansion process is usually performed as the mean vector of the state vector distribution [41].

The estimation process of EKF can be represented as equation 3.12. In this equation, G_t is the Jacobian of the nonlinear motion model function $f(\cdot)$ defined in equation 3.1 at the point of μ_{t-1}.

$$p(x_t|x_{t-1}) \approx \det(2\pi R_t)^{-\frac{1}{2}} exp\left\{-\frac{1}{2}[x_t - g(\mu_{t-1}) - G_t(x_{t-1} - \mu_{t-1})]^T R_t^{-1}[x_t - g(\mu_{t-1}) - G_t(x_{t-1} - \mu_{t-1})]\right\} \quad (3.12)$$

Similarly, the EKF measurement process can be represented as equation 3.13 where H_t is the Jacobian of the nonlinear measurement model function $h_t(\cdot)$ defined in equation 3.2 at the point of $\overline{\mu_t}$.

$$p(z_t|x_t) \approx \det(2\pi Q_t)^{-\frac{1}{2}} exp\left\{-\frac{1}{2}[z_t - h(\overline{\mu_t}) - H_t(x_t - \overline{\mu_t})]^T Q_t^{-1}[z_t - h(\overline{\mu_t}) - H_t(x_t - \overline{\mu_t})]\right\} \quad (3.13)$$

3.2.3 The Unscented Kalman Filter

The EKF releases the strong linear constraint of the Kalman filter by introducing a linearization operation in each state transition process. The most significant limitation of the EKF is that the quality of linearization operation (through first order Taylor expansion) heavily relies on the local linearity of the (nonlinear) state transition functions at the expansion point.

The Unscented Kalman Filter (UKF) [41] solves the problem of the unrealistic linear assumption in the Kalman filter by using a different approach. Instead of using linearization to maintain the Gaussian property of the belief distribution like EKF, the UKF achieves the same goal by reconstructing the Gaussian presentation of the belief distribution, namely the mean vector μ_t and the covariance matrix Σ_t, from results of the "sigma points" [41] transferred through nonlinear state transition functions.

The "sigma points" in UKF are selected according to deterministic algorithms. For an n-dimensional Gaussian with mean vector μ and covariance matrix Σ, there will be $2n + 1$ sigma points selected based on the rules represented in equation 3.14:

$$\mathcal{X}^{[0]} = \mu$$
$$\mathcal{X}^{[i]} = \mu \mp (\sqrt{(n+\lambda)\Sigma})_i \tag{3.14}$$

In this equation, λ is a scaling parameter that determines the distance that the sigma points are spread from the mean vector μ. The subscript i denotes the ith row of column of a square root matrix.

Each “sigma point” in the UKF also has a weight associated with it and this weight is used when re-establishing the mean and covariance of the estimated Gaussian belief distribution from the transferred “sigma points”. The weight ω of a “sigma point” is calculated as equation 3.15:

$$\omega^{[0]} = \frac{\lambda}{n + \lambda}$$
$$\omega^{[i]} = \frac{1}{2(n + \lambda)} \tag{3.15}$$

The state transition process of these selected “sigma points” is presented in equation 3.16 as:

$$\mathcal{Y}^{[i]} = g(\mathcal{X}^{[i]}) \tag{3.16}$$

Furthermore, the process of reconstructing a Gaussian belief distribution based on the transition results of the “sigma points” is presented in equation 3.17 as:

$$\mu' = \sum_{i=0}^{2n} \omega^{[i]} \mathcal{Y}^{[i]} \tag{3.17}$$

$$\Sigma' = \sum_{i=0}^{2n} \omega^{[i]} (\mathcal{Y}^{[i]} - \mu')(\mathcal{Y}^{[i]} - \mu')^T$$

A single iteration of the UKF is illustrated in figure 3.2. The input parameters of the UKF are the posterior distribution of the state vector from the previous iteration and the latest observation respectively. The return value of the UKF is the posterior distribution of the state vector at present.

As shown in figure 3.2, a set of "sigma points" are selected from the previous posterior distribution (μ_{t-1}, Σ_{t-1}) according to the algorithm presented in equation 3.14 and these sigma points are kept in $\mathcal{X}_{t-1}$. Then, the prior distribution of the present state vector is estimated by applying the prediction model to the selected sigma points and kept in $\overline{\mathcal{X}_t^*}$. As shown in figure 3.2, the mean vector $\overline{\mu_t}$ and variance matrix $\overline{\Sigma_t}$of the prior distribution are re-established from the transformed sigma points in $\overline{\mathcal{X}_t^*}$.

As shown in the green colour section of the UKF pseudo code in figure 3.2, a new set of "sigma points" are selected from the prior distribution ($\bar{\mu}_t$, $\bar{\Sigma}_t$) and then transformed through the observation model $h(\cdot)$. The mean vector $\hat{z}_t$ and the variance matrix S_t are re-established from the transformed sigma points in $\overline{\mathcal{Z}_t}$. Furthermore, the Kalman gain K_t is calculated from the variance matrix S_t and the co-variance matrix of both the estimated distribution approximated by $\overline{\chi_t}$ and the observed distribution approximated by $\overline{\mathcal{Z}_t}$.

Finally, as shown in the blue colour section in figure 3.2, the posterior distribution of the timestamp t is calculated from both estimation and observation by integrating the Kalman gain.

Unscented Kalman Filter:

Input parameters: μ_{t-1}, Σ_{t-1}, z_t; *Return values*: μ_t, Σ_t

BEGIN

$$\chi_{t-1} = (\mu_{t-1},\ \mu_{t-1} + \gamma\sqrt{\Sigma_{t-1}},\ \mu_{t-1} - \gamma\sqrt{\Sigma_{t-1}})$$

$$\overline{\chi_t^*} = g(\chi_{t-1})$$

$$\overline{\mu_t} = \sum_{i=0}^{2n} \omega_m^{[i]} \overline{\chi_t^*}^{[i]}$$

$$\overline{\Sigma_t} = \sum_{i=0}^{2n} \omega_c^{[i]} (\overline{\chi_t^*}^{[i]} - \overline{\mu_t})(\overline{\chi_t^*}^{[i]} - \overline{\mu_t})^T + R_t$$

$$\overline{\chi_t} = (\overline{\mu_t},\ \overline{\mu_t} + \gamma\sqrt{\overline{\Sigma_t}},\ \overline{\mu_t} - \gamma\sqrt{\overline{\Sigma_t}})$$

$$\overline{Z_t} = h(\overline{\chi_t})$$

$$\hat{z}_t = \sum_{i=0}^{2n} \omega_m^{[i]} \overline{Z_t}^{[i]}$$

$$S_t = \sum_{i=0}^{2n} \omega_c^{[i]} \left(\overline{Z_t}^{[i]} - \hat{z}_t\right)\left(\overline{Z_t}^{[i]} - \hat{z}_t\right)^T + Q_t$$

$$\overline{\Sigma_t}^{x,z} = \sum_{i=0}^{2n} \omega_c^{[i]} (\overline{\chi_t}^{[i]} - \overline{\mu_t})(\overline{Z_t}^{[i]} - \hat{z}_t)^T$$

$$K_t = \overline{\Sigma_t}^{x,z} S_t^{-1}$$

$$\mu_t = \overline{\mu_t} + K_t(z_t - \hat{z}_t)$$

$$\Sigma_t = \overline{\Sigma_t} - K_t S_t K_t^T$$

RETURN μ_t, Σ_t

END

Figure 3.2: The pseudo code of UKF [37].

3.3 Particle Filter

The Kalman filter family discussed in the previous section has a common limitation: the belief distribution of the state vector is restricted to be a Gaussian distribution. However, this constraint is usually too restrictive for human tracking applications in unconstrained environment.

The particle filter eliminates the above restriction by presenting the belief distribution of a state vector in a nonparametric manner with a set of Monte Carlo samples (particles). In this way, a much larger range of distributions, rather than merely Gaussian distribution, can be presented properly. Furthermore, the belief distribution of a state vector can also be a mixture of multiple distributions in the particle filter. Therefore, the particle filter is able to track multiple hypothesis of a target at the same time and postpone the false positive error elimination process naturally after more observations are available.

Similar to the UKF, the particle filter performs state transition through a set of Monte Carlo samples and therefore eliminates the requirement of linearization operations performed in EKF.

The Monte Carlo samples used in a particle filter are denoted as $\mathcal{X}_t := x_t^{[1]}, x_t^{[2]}, \dots, x_t^{[M]}$ where each particle $x_t^{[m]}, 1 \leq m \leq M$ is a concrete instantiation of the

state vector. Each particle is associated with a weight $\omega_t^{[m]}$ to indicate the importance of a particular Monte Carlo sample in a belief distribution.

A single iteration of the particle filter is illustrated in figure 3.3. The input parameters of the particle filter are the posterior belief distribution of the state vector in the previous time step (represented by a set of weighted Monte Carlo samples) and the observation vector of the current time step. The return value is the posterior belief distribution of the current time step, which is also represented by a set of weighted Monte Carlo samples.

Particle Filter:

Input parameters: $\mathcal{X}_{t-1}$, z_t; *Return values:* $\mathcal{X}_t$

BEGIN

$\overline{\mathcal{X}_t} = \mathcal{X}_t = null$

FOR $m = 1$ **TO** M **DO**

$\overline{x}_t^{[m]} \sim p\left(x_t \middle| x_{t-1}^{[m]}\right)$

$\overline{\mathcal{X}_t} \leftarrow \overline{x}_t^{[m]}$

ENDFOR

FOR $m = 1$ **TO** M **DO**

$\omega_t^{[m]} = p\left(z_t \middle| \overline{x}_t^{[m]} \in \overline{\mathcal{X}_t}\right)$

$\mathcal{W}_t \leftarrow \omega_t^{[m]}$

ENDFOR

FOR $m = 1$ **TO** M **DO**

Draw $x_t^{[i]} \in \overline{\mathcal{X}_t}$ with probability $\propto \omega_t^{[i]} \in \mathcal{W}_t$

$\mathcal{X}_t \leftarrow x_t^{[i]}$

ENDFOR

RETURN $\mathcal{X}_t$

END

Figure 3.3: The pseudo code of particle filter [37].

As shown in figure 3.3, the pseudo code of the particle filter is divided into three colour coded sections based on their functions. The estimation process is performed in the red

colour section in figure 3.3. As shown, particles are drawn from the posterior belief distribution $\mathcal{X}_{t-1} := x_{t-1}^{[1]}, x_{t-1}^{[2]}, \dots, x_{t-1}^{[M]}$ based on the state transition probability $p(x_t|x_{t-1})$ of the estimation step defined in equation 3.3. The weights of these particles are uniformly distributed because the current observation has not being incorporated yet. The (unweighted) distribution of these particles presents the prior belief distribution of the state vector in the system, i.e. $\overline{bel(x_t)} \sim \overline{\mathcal{X}_t} := \overline{x_t}^{[1]}, \overline{x_t}^{[2]}, \dots, \overline{x_t}^{[M]}$.

The measurement process defined in equation 3.4 is performed in the green section in figure 3.3. As shown, each particle distributed follows the prior distribution $\overline{bel(x_t)}$ and weighted by incorporating the current observation vector. The posterior belief distribution of the state vector $bel(x_t)$ is presented by the weighted distribution of these particles because of the definition presented in equation 3.4.

The resampling process illustrated in the blue section in figure 3.3 is unique to the particle filter. It draws M new particles from the current particle set and the probability of drawing each particle is given by its weight. The purpose of this process is to prevent the problem of particle degeneration by transferring the unweighted particle distribution from approximating the prior belief $\overline{bel(x_t)}$ to approximating the posterior belief $bel(x_t)$. A branch of resampling algorithms are proposed in the literature, which caters for different application specific features. Surveys of resampling algorithms in particle filters can be found in [43-48].

3.4 Summary

In computer vision tracking using stochastic filtering, the accuracy of the Kalman filter relies heavily on the linearity of the system. However, being linear is a strong constraint and is usually unrealistic in an object tracking system. The extended Kalman Filter (EKF) improves the Kalman filter by relaxing this linear constraint. However the most significant limitation of the EKF is that the quality of linearization operation (through first order Taylor expansion) relies heavily on the local linearity of the (nonlinear) state transition functions at the expansion point. The unscented Kalman Filter (UKF) solves the problem of the unrealistic linear assumption in the Kalman filter. All these Kalman filters have two common limitations: the belief distribution of the state vector is assumed to be Gaussian and only a single candidate state can be represented. However, these assumptions are too restricted to properly model the belief distributions in real world systems, especially in the object tracking domain.

The particle filter eliminates these constraints by presenting the belief distribution of a state vector in a nonparametric method with a set of Monte Carlo samples (particles). In this way, a much larger range of distributions is used, rather than merely Gaussian.

Chapter 4
Metaheuristic particle filters in Object Tracking

This chapter discusses some metaheuristic optimisation algorithms that have been applied in particle filter based tracking applications. Metaheuristics is an optimisation method that allows larger scale problems to be tackled in a reasonable amount of time although there is no guarantee for metaheuristics to find the global optimal solution.

The metaheuristic method was originally introduced into vision tracking systems in [49]. In this research, a modified simulated annealing algorithm was embedded into a particle filter to improve the accuracy of a full body motion capture application. Thereafter, research interests in integrating metaheuristics with particle filters started to emerge and more metaheuristic algorithms have been explored.

Despite the amount of research in this area, most of the proposed algorithms are focused on full body tracking applications, such as [5, 49-54]. This is because: 1) Full body motion capture research deals with the high degree of freedom present in human motion. Yet despite the large number of particles that an overall system may use, the average particles per degree of freedom can still be small. However, the generic particle

filters easily converge at a local minimum when using such small numbers of particles and consequently, tracking accuracy is reduced. 2) Full body motion tracking is usually performed in highly controlled environments, for example in a dedicated studio. In this case, the tracking target is always represented by the global optima, which provides an ideal platform for metaheuristic optimisation. 3) Real time processing speed is not required in most of these earlier full body motion capture projects.

As a consequence, there is a strong need for optimisation algorithms to be applied to overcome the significant time penalty required for good solutions to merge. This chapter provides a review of some metaheuristic algorithms and their applications in computer vision based tracking research. The discussion focuses on the optimisation algorithms that have been integrated with the particle filter in prior research, such as the evolutionary algorithm, the scatter search algorithm, the path relink algorithm, the particle swarm algorithm and the simulated annealing algorithm.

In this chapter, the term ‘solution’ is used to indicate a possible position of the target in a tracking system and a ‘good’ solution means that the position indicated by ‘this’ solution is close to the actual position of the target.

4.1 Metaheuristic Algorithms for Particle Filter Optimisation

4.1.1 Evolutionary Algorithm

The evolutionary algorithm is a population based metaheuristic algorithm that simulates the natural process of a species adapting to the environment [55]. The pseudo code in figure 4.1 shows the main procedure of the evolutionary algorithm in a generalized form. In the pseudo code, $P_{initial}$ indicates the initial population; $P_{current}$ indicates the population at the current iteration; P_{parent} indicates a set of selected solutions from $P_{current}$; and $P_{children}$ indicates generated population from P_{parent}. As shown, the algorithm starts with a set of tentative solutions, which are usually generated by random selection. Then, these initial solutions are evaluated by a predefined objective function to associate a fitness value with every individual solution before they are passed to an iterative optimisation process that contains four steps: selection, generation, evaluation and replacement.

Evolutionary Algorithm:

Return values: the best individual found

BEGIN

Generate initial population: $P_{initial}$

Assign $P_{initial}$ to $P_{current}$

Evaluate the population $P_{current}$

WHILE (Termination criterion NOT meet) **DO**

Select a subset of solutions from $P_{current}$ to $P_{parents}$

Produce a new population $P_{children}$ from $P_{parents}$

Evaluate the new population $P_{children}$

Replace $P_{current}$ with selected solutions in $P_{children}$

ENDWHILE

RETURN The best individual in $P_{current}$

END

Figure 4.1: Pseudo code of the evolutionary algorithm in a generalized form.

In the selection step, a subset of solutions is chosen from the current population $P_{current}$ as parents for offspring to new solutions. The principle of this selection step in the evolutionary algorithm is similar to the resampling step in the particle filtering process introduced in the last chapter, which is that optimal solutions shall have more probability to be selected and the solutions with low fitness values shall also have some small chances to be selected as well.

In the generation process, a set of new solutions, $P_{children}$ are generated based on the selected parent solutions. Crossover and mutation are the two commonly used methods to generate new solutions. The crossover process involves two or more parent solutions. The characteristics of the parent solutions are recombined to generate an offspring solution. The offspring solution is expected to lead the searching process towards a 'better' area because it inherits features from its parents that are most likely selected from elite solutions in the previous iteration. There are many approaches to performing crossover optimised for application objectives. For example, in computer vision based tracking, the intermediate crossover method or the two-point crossover method have proven to be better choices than the order crossover method [55].

The mutation process is a unary operation. It represents a random small change in a solution, which is usually applied to a small portion of the offspring solutions. Similar to the crossover process, the method used for mutation operations is chosen based on its area of application. For example, if the solution is presented in binary form, then a bit flip operation can be a good candidate for mutation. However, in computer vision tracking applications a better choice could be a uniform / Gaussian random mutation because the binary mutation method is more likely to generate infeasible solutions.

The evaluation process followed by the generation step is to evaluate the generated solutions with an objective function so that each new solution will be associated with a

fitness value. The fitness value is one of the contributing factors deciding whether a solution survives a particular iteration or not.

The replacement process eliminates unfit solutions from the overall population to maintain the constant size of the $P_{current}$ population. There are different types of replacement methods proposed in the literature, ranging from generational replacement in which the whole offspring population will replace the parent population systematically to steady-state replacement in which only one offspring solution is chosen to replace the worst solution in the parent population. In general, if elitism is favourable in the replacement process, then the overall optimisation will lean towards fast convergence at the risk of highly premature results, whereas if egalitarianism is favourable in this process, the overall algorithm will lean towards slow convergence but this is likely to generate a better solution.

There exists a number of methods to combine the particle filter with the evolutionary algorithm, such as the genetic filter described in [56] and the evolutionary particle filter in [55]. The principles of both algorithms are similar, in that the resampling stage of the particle filtering process is replaced by a partially implemented evolutionary optimisation to address the "deprivation problem" [56]. The same problem is referred as the "impoverishment problem" in [55]. Apart from their similarity in the fundamental structure, the differences between these two methods are in the genetic filter, both crossover operation and mutation operation are used. However, the evolutionary algorithm is only

performed once, rather than an iterative process as introduced earlier, in the genetic filter. On the other hand, in the evolutionary particle filter [55], the mutation operation is omitted totally. Only the crossover operation is performed with selected particles. The number of iterations for the evolutionary algorithm is predefined and remains the same throughout the whole tracking process.

4.1.2 Scatter Search and Path Relink

Scatter search and path relink algorithms are two population-based metaheuristic optimisation algorithms based on similar frameworks. In the scatter search / path relink algorithm, there exists two phases, diversification and intensification. The diversification phase is the step that the reference set (in which the searching process is performing) is selected in a diverse manner from the initial solution space. The intensification phase is the step in which the solution space (represented by the reference set) is explored as much as possible.

Figure 4.2 shows a generalized form of the scatter search / path relink algorithm. As shown, a set of highly diverse solutions are generated at the beginning of this algorithm. Then, a set of reference population is selected from the initial solutions, which forms a workspace for the present iteration. The reference population is selected in a diversification manner so that the search can cover as much area in the initial solution space as possible. The selected reference set is then processed by the inner loop, where the intensification phase of the algorithm is implemented. In this phase, a sub

population (P_{sub}) is chosen from the reference set based on an elitism method. Then, a set of new solutions is generated from P_{sub} through some recombination strategies. In the scatter search algorithm, the new solutions are usually generated by interpolating middle values between a pair of existing solutions in the P_{sub} whereas in the path relink algorithm, the new solutions are usually generated by interpolating trajectories that link between a pair of existing solutions in the P_{sub}. The generated solutions are evaluated and improved as non-related individuals through a simple optimisation algorithm, such as local search. Finally, the reference set is updated through an elitism approach and the intensification search phase terminates when the solutions in the reference set remain stable (converged).

If the termination criterion of the optimisation algorithm is not satisfied (outer loop), a new set of reference solutions will be reselected from the initial solution space. The new reference solutions are then used to continue the search process for the next iteration.

Scatter Search / Path Relink Algorithm:

Return values: the best individual found

BEGIN

Generate highly diverse set of initial population $P_{initial}$

Evaluate the population $P_{initial}$

WHILE (Termination criterion NOT meet) **DO**

Select reference set $P_{reference}$ from $P_{initial}$ with diversification strategy.

REPEAT

Select subsets P_{sub} from $P_{reference}$

Generate $P_{combination}$ by apply recombination strategy on P_{sub}

Improve $P_{combination}$ by improving each element in $P_{combination}$ with single metaheuristic method

Update the reference set $P_{reference}$ with the improved $P_{combination}$

UNTIL ($P_{reference}$ converge)

Regenerate $P_{current}$ based on $P_{reference}$.

ENDWHILE

RETURN The best individual

END

Figure 4.2: Pseudo code of the scatter search and path relink algorithm in a generalized form.

Figure 4.3 [54] illustrates the processing steps of the intensification phase discussed above in both the scatter search algorithm and the path relink algorithm. As shown, the main differences of the two algorithms appear at the stage when the new solutions are generated from the reference set. In the path relink algorithm (the upper half in figure

4.3), the new solutions are generated by interpolating the trajectories between selected solutions whereas in the scatter search algorithm (the lower half in figure 4.3), the new solutions are generated by interpolating the middle solution of the existing solution pairs.

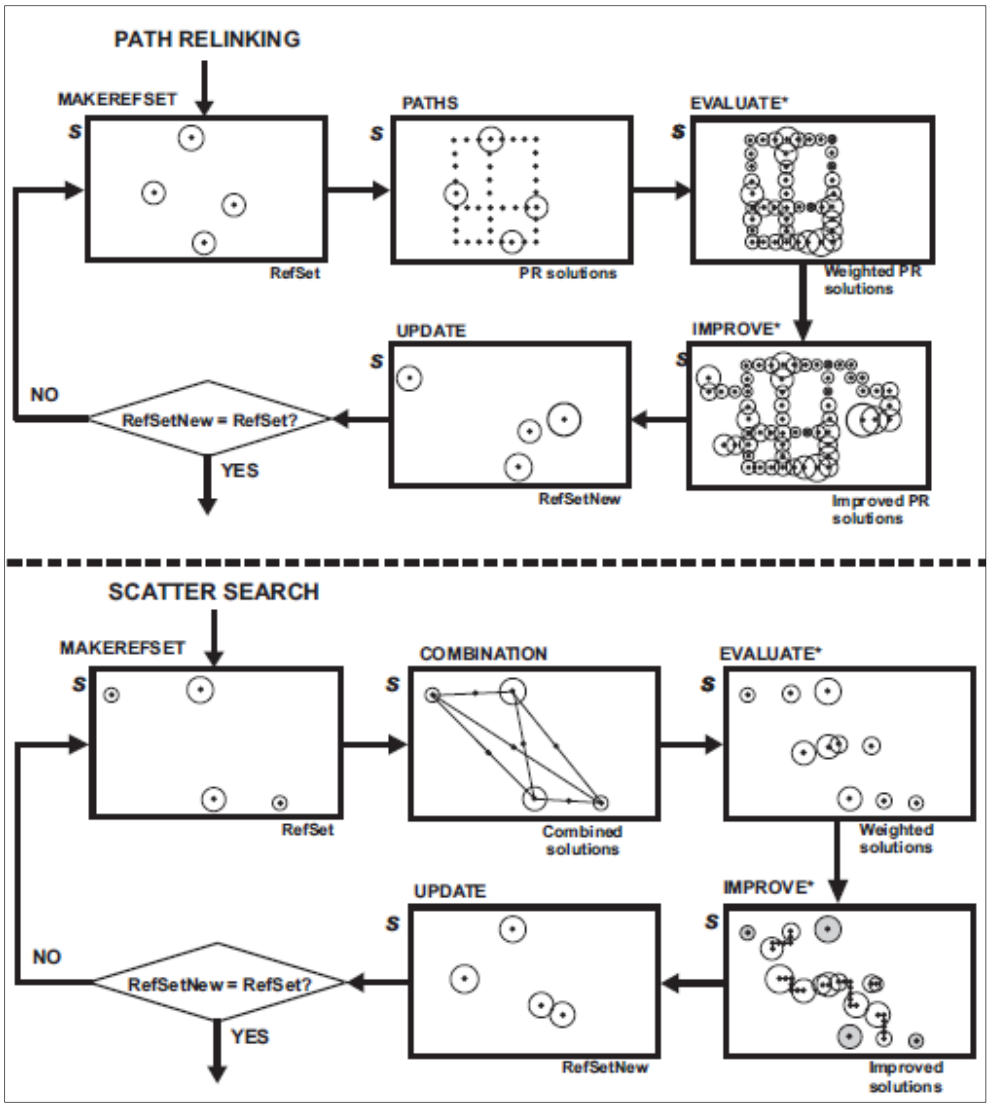


Figure 4.3: A comparison between the scatter search algorithm and the path relink algorithm in the intensification phase [54].

Scatter search particle filter and path relink particle filter are both described in [54]. In this research, the optimisation algorithm is embedded into particle filters and the posterior particles generated by the particle filter are defined as the initial solutions to be optimised. The optimised particles are then used to update the particle population in the particle filter so that the posterior distribution of the particles can approximate the target distribution more accurately. Then the particles are resampled and passed to the state transition model for the next iteration.

4.1.3 Particle Swarm Algorithm

In the particle swarm algorithm, each individual particle has its own dynamic configuration (velocity vector), which is influenced by both the global optimal at the present time step and the best known solution along the particle's trajectory.

Figure 4.4 illustrates the pseudo code of the particle swarm algorithm. The particle swarm optimisation process starts with a randomly generated initial population. Each particle contains two main factors: a state vector that represents a tentative solution and a velocity vector that defines the amount of change that will be applied to this particle. As shown in figure 4.4, in each iteration step, the velocity vectors of the particles are updated based on the latest knowledge of the global optima and the best history in this particle's trajectory. Equation 4.1 illustrates a common method of updating the particle velocity in the particle swarm algorithm. In this equation, factor ω is called 'inertia weight'

[57] and it controls the impact of the previous velocity $v_i(t-1)$ on the current velocity $v_i(t)$. There are two random factors in this equation, namely ρ_1 and ρ_2. They are called learning factors, which represent the attraction that a particle has either toward its own success (p_{p-best}) or toward the global optima (p_{g-best}).

$$v_i(t) = \omega \times v_i(t-1) + \rho_1 \times \left(p_{p-best} - x_i(t-1)\right) + \rho_2 \times \left(p_{g-best} - x_i(t-1)\right) \quad (4.1)$$

As illustrated in the pseudo code, the particle's state vector is updated with the new velocity vector and then the new particle is evaluated with the fitness function defined in the system. If the new particle's fitness value is better than the global optima, then this particle will become the new global best (p_{g-best}). Similarly, if the fitness value at present is better than all fitness values in the history of this particle, then the local best (p_{p-best}) is updated with the new one.

Particle Swarm Algorithm:

Return values: the best individual found

BEGIN

Generate initial swarm population: $P_{initial}$

Assign $P_{initial}$ to $P_{current}$

Evaluate the population $P_{current}$

WHILE (Termination criterion NOT meet) **DO**

FOREACH (Particle p_i in the $P_{current}$) **DO**

Update the particle's velocity with equation 4.1

Update the particle's position with the new velocity.

Evaluate the particle p_i

If p_i is better than the global best p_{g-best} then update p_{g-best}

If p_i is better than its own past p_{p-best} then update p_{p-best}

ENDFOREACH

ENDWHILE

RETURN The best individual p_{g-best}

END

Figure 4.4: Pseudo code of the particle swarm algorithm in a generalized form.

The particle swarm optimisation is also being used to improve the performance of the particle filter. Examples of such combinations can be found in [57], [58] and [59]. The logic structures used in these algorithms are similar in general. That is, the particles are propagated by the motion model defined as part of the ordinary particle filter. Then, after

these particles are evaluated by the observation model in the particle filter, they are optimised according to the particle swarm iteration (figure 4.4).

4.1.4 Simulated Annealing

The simulated annealing algorithm simulates the metallurgy process of obtaining a strong crystalline structure, which requires heating and then slowly cooling a substance. In this process, a high initial temperature and slow cooling process lead to a stronger crystalline structure as whereas a low initial temperature and fast cooling process leads to imperfections. In the latter case, the cooling substance will not attain thermal equilibrium [60]. The generalized framework of this algorithm is illustrated in figure 4.5. As shown, the optimisation process contains two phases: the cooling phase and the equilibrium phase. The simulated annealing based search for the global maximum of a Gibbs probability distribution has generally the exponential time complexity.

The cooling phase is implemented in the outer loop in the pseudo code (figure 4.5). In this phase, the temperature, which is a term borrowed from its metallurgy origin, decreases according to a predefined cooling algorithm. The concept of temperature controls the searching behaviour of the simulated annealing algorithm. If the temperature is very high, which happens when the optimisation process is just started, then the algorithm is more or less corresponding to a random walk process in the whole solution space. On the other hand, when the temperature is very low, which usually

happens when the optimisation process is about to finish, then the algorithm is corresponding to a hill climbing algorithm.

The equilibrium phase is implemented in the inner loop in the pseudo code (figure 4.5). As long as the equilibrium condition is not satisfied, a new solution will be generated from the current solution. Then, the new solution's fitness value will be evaluated and compared with the original solution's fitness value. If the new solution is better from the perspective of the objective function, then it will be accepted as the new current solution that will be used in the next iteration. However, if the new solution is not better than the existing one, it still has a chance to be accepted. The probability of a non-improving move being accepted is related to the temperature value. The higher the current temperature, the more likely the new non-improving solution will be accepted. The method of calculating accept solution in the pseudo code is called: Metropolis sampler [60].

Simulated Annealing Algorithm:

Return values: the best solution found

BEGIN

Generate an initial solution and assign it to $S_{current}$.

Assign initial temperature T_{max} to $T_{current}$

WHILE ($T_{current} > T_{min}$ and termination criterion NOT meet) **DO**

WHILE (Equilibrium condition is NOT reached) **DO**

Generate a random neighbour $S_{neighbour}$ from $S_{current}$.

Evaluate and assign the fitness values to: $E_{neighbour}$ and $E_{current}$.

IF ($E_{neighbour} \geq E_{current}$) **THEN**

Accept the neighbour solution: $S_{current} = S_{neighbour}$

ELSE

Accept the neighbour solution with a probability $e^{\frac{-\Delta E}{T_{current}}}$

ENDIF

ENDWHILE

Update current temperature: $T_{current}$ based on a chosen cooling scheme.

ENDWHILE

RETURN The best solution found.

END

Figure 4.5: Pseudo code of the simulated annealing algorithm in a generalized form.

The simulated annealing algorithm in a particle filtering process was originally in [49] and has been explored extensively in the computer vision literature [51, 61-64]. The

simulated annealing algorithm is used in the tracking process to either avoid the particle "kidnapping" problem described in [61] when multiple particle filters are used simultaneously in a multi-face tracking system or to avoid the problem of particles being stuck at a local minima when the particle filter is used for complex tracking tasks [49, 63, 64].

In [49], the simulated annealing process is reinterpreted with two new terms: the term 'layer' in [49] refers to a similar concept as temperature in the simulated annealing process and the term 'annealing run' refers to the equilibrium iteration introduced in figure 4.5. In every time step of the annealing particle filter, an annealing run (equilibrium iteration) is started at the highest layer that corresponds to the high temperature in the simulated annealing. The initial solutions of the annealing run come from the resampled particles output from the higher layer where for the highest layer, the initial solutions are provided by the resampled posterior particles generated by the particle filtering process. To impose diversification, the particles are augmented with noises generated from a zero mean Gaussian distribution. The layer indexes are gradually reduced to zero to complete the annealing optimisation process at the current time step.

4.2 Summary

In this chapter, the metaheuristic algorithms that have been integrated with the particle filter in the literature are discussed. These algorithms are successful in their applications, such as full body motion capture in a constrained environment. However, their potential for real time human tracking in unconstrained environments is not so promising because 1) the assumption that the global optimal in the image is the target of interest and 2) they not designed for real time performance.

Chapter 5
Tabu Search Particle Filter

In this chapter, a new metaheuristic particle filter is described. This algorithm is based on the combination of the generic particle filter and a modified tabu search metaheuristic algorithm. The proposed algorithm is designed for real time tracking systems in unconstrained environments (for example cluttered backgrounds and non-stationary cameras).

The proposed filter overcomes the shortcomings of the existing algorithms by:

1. imposing a connection between the objective function and the target's motion model so that the evaluation process is not only with respect to the target's appearance model but also the target's motion history;
2. building a data structure to record all visited solutions so that duplicated visits can be avoided to improve performance;
3. refining the known optima and searching recently revealed regions at the same time so that the algorithm can converge in minimum time and still have the opportunity of correcting inaccurate prior estimates.

The tabu search metaheuristic algorithm is introduced in section 5.1. This is followed by the discussion of the tabu search particle filter in section 5.2.

5.1 Tabu Search Metaheuristic Optimization

Figure 5.1 illustrates a generalized form of the tabu search metaheuristic optimisation algorithm [65, 66]. As shown, the original tabu search algorithm is a single solution algorithm in which one solution is optimised at a time.

Tabu Search Algorithm:

Return values: the best solution found

BEGIN

Generate an initial solution and assign it to $S_{current}$.

Initialise the tabu list T.

WHILE (Termination criterion NOT meet) **DO**

Find the best admissible neighbor solution $S_{best-neighbor}$

Assign $S_{best-neighbor}$ to $S_{current}$.

Update the tabu list T.

IF (Meet intensification criterion) **THEN**

Apply intensification method.

ENDIF

IF (Meet diversification criterion) **THEN**

Apply diversification method.

ENDIF

ENDWHILE

RETURN The best solution found.

END

Figure 5.1: Pseudo code of the tabu search algorithm in a generalized form.

As shown in figure 5.1, an initial solution is generated in the solution space when the tabu search process is started. The performance of the optimisation process and the quality of the optima found through this process is affected by the selected initial solution, which is chosen based on the prior knowledge of the landscape structure of the problem.

After the initial solution is selected, the tabu list is prepared before the search process begins. The tabu list is a unique concept in the tabu search algorithm, which remembers recent solutions or moves. This prevents the searching process from cycling back to the previously visited solutions, especially when the search is moving away from local optima. The maximum length of the tabu list is a configurable parameter called the tabu tenure. During the search process, the last move (or solution) is added in the tabu list and, if the total length of the tabu list is more than the tabu tenure, the oldest record is removed from the list.

As shown in figure 5.1, at the beginning of each search iteration (inside the while loop), the best solution in the neighbourhood is selected, even if the new solution has not improved from the current one. The new solution will be admissible, which means that the new solution will not be in the tabu list. An exception to this rule is that if a predefined aspiration criterion is satisfied, then a solution is treated as admissible even if this solution or the move to generate this solution is listed in the tabu list. The aspiration criterion can improve the performance of the tabu search process in some

applications. For example, an otherwise tabued move will be accepted if it will generate a solution that is better than all solutions obtained so far. As in the search process, the tabu list is updated by appending the new solution to the tabu list and the oldest record removed if the total length of the tabu list is more than the predefined tabu tenure.

The intensification and diversification are used by the tabu search algorithm to improve the performance of the optimisation process. The intensification process encourages the searching process to explore more thoroughly in the regions where good solutions have been found in the past. The diversification process prevents the search process from wasting time in regions where the optima are unlikely to be found.

5.2 The Proposed Tabu Search Particle Filter

As discussed previously, the metaheuristic particle filters proposed in the literature are difficult to be applied in real time tracking applications in unconstrained environments. These algorithms are not designed for real time application and they assume the target of interest is represented by the global optimum in the whole image. In this research, a unique metaheuristic particle filter has been developed based on the generic particle filter and an improved tabu search optimisation algorithm. The proposed tabu search particle filter overcomes the problems of the metaheuristic particle filters in previous research by: 1) imposing an extra constraint to the objective evaluation function so that the target's previous motion history is taken into account during the optimisation

process; 2) reducing the time used for the optimisation process by preventing a solution space being revisited during the same iteration; 3) binding the decision of intensification / diversification to the area of the solution space being covered in the searching process, which improves the convergence speed of the optimisation process.

Figure 5.2 illustrates the pseudo code of the new tabu search particle filter. To simplify the discussion, the estimation and measurement processes that are identical to the generic particle filter are excluded from the pseudo code.

As shown in figure 5.2, the optimisation process in the new algorithm is population based and the initial solutions come from the posterior distribution of the target's state vector ($bel(x_t)$), which is represented by a group of weighted particles: $\mathcal{X}_t := x_t^{[1]}, x_t^{[2]}, \dots, x_t^{[M]}$. The tabu list T is initialised by the state vector of these particles and their corresponding fitness values. In this algorithm, the tabu tenure is 'indefinite', which means that all visited solutions are recorded in the tabu list and these records will not be removed from the tabu list until the optimisation process is completed.

Tabu Search Particle Filter (the optimization part):

Return values: the best individual found

BEGIN

Generate the initial solution set $P_{initial}$ from the posterior distribution $bel(x_t)$.

Initialise the tabu list T with solutions in $P_{initial}$ and their fitness values.

WHILE (Termination criterion NOT meet) **DO**

Divide all solutions in $P_{current}$ into three categories based on their fitness values. The P_{good} set contains solutions with high fitness values, the P_{poor} set contains solutions with low fitness values and the P_{normal} set contains the rest solutions in the $P_{current}$.

► **Search Process for the 'Normal' Particles**

► **Search Process for the 'Poor' Particles**

► **Search Process for the 'Good' Particles**

Merge P_{good}, P_{normal} and P_{poor} into the new $P_{current}$

ENDWHILE

RETURN The best individual in $P_{current}$

END

Figure 5.2: Pseudo code of the proposed tabu search particle filter.

At the beginning of each iteration, the whole population of the particles is divided into three categories based on their fitness values. The solutions with the best fitness values are classified into the 'good' particle set that is represented by P_{good} in figure 5.2. These particles are elite solutions in the system, which will guide the intensification process later on. The number of elite solutions in P_{good} is limited to a small quantity, for example 10 particles at most or less than 10 percent of the overall particle population, because of

the single-target-per-tracker assumption in this research. On the other hand, the worst fitting solutions in the system are classified into the ‘poor’ particle set during this process. These particles represent the most unlikely regions where the optima could be found. The poor particles in this category are diverted to other areas in the solutions. Finally, the remaining particles that do not belong to either the ‘good’ particle set or the ‘poor’ particle set are grouped together and classified as the ‘normal’ particle set. Particles in this category have medium fitness values, which indicates the solution space around these particles are worth being investigated by the optimisation process to look for better optima. The performance of the optimisation process is affected by the ratio of the number of particles in the ‘poor’ and ‘normal’ categories. Given a fixed number of particles, quicker but possibly premature convergence is usually achieved by a larger ‘poor’ population whereas a thorough search in the whole solution space is usually achieved by a large population in the ‘normal’ particle set. In this research, a unique ‘punishment’ strategy is applied to the particles belonging to the ‘normal’ category, which allows the ratio between the ‘normal’ population and the ‘poor’ population to dynamically adapt to the searching process.

As shown in figure 5.2, the particles are processed with different strategies according to their categories in the sequence of ‘normal’ particles, ‘poor’ particles and ‘good’ particles. The details of these search processes are illustrated in figure 5.3, 5.4 and 5.5, which will be discussed later. Finally, after all particles in three categories are processed, these

particles are remerged into $P_{current}$ and passed to the next iteration for further optimisation.

The optimisation strategy for the particles that belong to the 'normal' category is illustrated in figure 5.3. As shown, a new particle is generated by replacing an existing particle with additive zero mean Gaussian random noise: $\mathcal{N}(0, \sigma_{normal})$. Then, the tabu list is checked with the state value in the new particle. If the solution space represented by the new particle has not been visited before, the particle is evaluated with the objective function and then the tabu list is updated with this new solution. If the new particle represents a solution that has been visited before, then the aspiration condition is tested to decide whether the move will be approved (aspiration) or rejected (tabu). The first aspiration condition designed in this research is to check whether the fitness value of the new solution (the one that has been visited by other particles before) is better than all solutions that this particle has visited in all previous iterations. If so, then the move is exempt from being tabued because it improves the best fitness value of this particle. Otherwise, the move is rejected and the state vector of the particle is rolled back to its previous value. Furthermore, the particle will be penalised for making a tabued move by downgrading it from the 'normal' category to the 'poor' category.

The decision of using the diversification strategy or the intensification strategy in the search process is made at run time based on the hit rate of the tabu list. As the optimisation process evolves through iterations, the chances that a 'normal' particle makes a tabued move are increased because the tabu list covers more of the solution

space. The search emphasis is naturally shifted from exploring the whole solution space to avoid being trapped in the local optima and to refining the search in optima currently found in the search space so that the algorithm can converge quickly.

Search Process for the 'Normal' Particles

BEGIN

FOREACH (Solution s in P_{normal}) **DO**

Generate and replace solution $\hat{s} = s + \dot{n}(0, \sigma_{normal})$.

IF ($\hat{s}$ is NOT in the tabu list T) **THEN**

Evaluate the fitness of $\hat{s}$.

Insert $\hat{s}$ (and its fitness value) in the tabu list T.

ELSE

Retrieve the fitness value of $\hat{s}$ from the tabu list T.

IF (the fitness of $\hat{s}$ is better than the fitness of s) **THEN**

Approve the move: $s \rightarrow \hat{s}$ (Aspiration condition)

ELSE

Reject the move: $s \rightarrow \hat{s}$ (Tabu condition)

Remove the solution from P_{normal} to P_{poor} (punishment)

ENDIF

ENDIF

ENDFOREACH

END

Figure 5.3: Pseudo code of the searching strategy for the 'normal' particle population in the proposed tabu search particle filter.

The optimisation strategy for the particles that belong to the 'poor' category is illustrated in figure 5.4. As shown, each particle in the 'poor' category is redistributed around an elite solution selected in sequence from the 'good' particles' category. The distance between the poor particle and the corresponding elite solution is controlled by a random factor drawn from a zero mean Gaussian distribution with predefined standard derivation σ_{poor}. Usually the σ_{poor} variable is chosen to be a small value so that the neighbourhood area of the elite solution can be thoroughly searched by these poor particles. As shown in figure 5.4, the move in this process is guaranteed to be accepted regardless of whether the generated solution is on the tabu list or not. This strategy is designed based on the assumptions that the neighbourhood of an elite solution will be thoroughly searched through intensification and that the computation cost to find an unvisited solution space in this area is more expensive than just allowing a tabued visit. This is the second aspiration rule designed in this research, that is: a tabued move is aspirated if the particle belongs to the 'poor' category.

The diversification strategy in this algorithm is designed to move particles away from its original solution space if the original fitness value is low enough to be classified as a 'poor' particle. At the same time, because these particles are redistributed in the neighbourhood of elite solutions, the search around optima are intensified at the same time by allocating more resource (particles in this context) to this region.

As show in figure 5.4, if a new particle generated in the surrounding area of an elite solution exists in the tabu list, its fitness value can be retrieved immediately from the tabu list without performing a redundant evaluation through the objective function. This strategy improves the search speed of the proposed algorithm.

Search Process for the 'Poor' Particles

BEGIN

FOREACH (Solution s in P_{poor}) **DO**

Select (in sequence) a solution s_{good} from 'good' particles: P_{good}.

Generate and replace solution $\hat{s} = s_{good} + \dot{n}(0, \sigma_{poor})$.

IF ($\hat{s}$ is NOT in the tabu list T) **THEN**

Evaluate the fitness of $\hat{s}$.

Insert $\hat{s}$ (and its fitness value) in the tabu list T.

ELSE

Retrieve the fitness value of $\hat{s}$ from the tabu list T.

ENDIF

ENDFOREACH

END

Figure 5.4: Pseudo code of the searching strategy for the 'poor' particle population in the proposed tabu search particle filter.

The optimisation strategy for particles that belong to the 'good' category is illustrated in figure 5.5. A new particle is generated by replacing an existing particle with additive zero mean Gaussian random noise: $\mathcal{N}(0, \sigma_{good})$. A small standard derivation is assigned to σ_{good} in this process so that the generated solution is in the surrounding

neighbourhood of the original elite solution. The tabu list is then checked using the state value of the new particle.

As shown in figure 5.5, if the new particle represents a solution that has not been visited before, the fitness value of this solution is calculated based on the objective function. However, unlike the search process for 'normal' particles, the strategy used for the 'good' particle is that the move is allowed if and only if the new solution is better than the original one. This strategy prevents the elite particles moving towards non-improving solution spaces.

Alternatively, if the new solution generated from this elite particle has been evaluated before, then the fitness value recorded in the tabu list is used to compare with the fitness value of the original solution. The move is approved when the new solution is better than the old one. Otherwise, the move will be rejected and the particle remains unchanged for this iteration.

Fitness thresholds and variances of Gaussian are selected manually based on intensive experimental evaluations. These variables are influenced by the following factors: image dimension, motion mode (target dynamic features), observation model (the shape of the observation distribution, i.e. the slope is high or low).

Search Process for the 'Good' Particles

BEGIN

FOREACH (Solution s in P_{good}) **DO**

Generate and replace solution $\hat{s} = s + \dot{n}(0, \sigma_{good})$.

IF ($\hat{s}$ is NOT in the tabu list T) **THEN**

Evaluate the fitness of $\hat{s}$.

Insert $\hat{s}$ (and its fitness value) in the tabu list T.

IF ($\hat{s}$ is better than s) **THEN**

Approve the move: $s \rightarrow \hat{s}$

ELSE

Reject the move: $s \rightarrow \hat{s}$

ENDIF

ELSE

Retrieve the fitness value of $\hat{s}$ from the tabu list T.

IF (the fitness of $\hat{s}$ is better than the fitness of s) **THEN**

Approve the move: $s \rightarrow \hat{s}$ (Aspiration condition)

ELSE

Reject the move: $s \rightarrow \hat{s}$ (Tabu condition)

ENDIF

ENDIF

ENDFOREACH

END

Figure 5.5: Pseudo code of the search strategy for the 'good' particle population in the proposed tabu search particle filter.

5.3 Summary

A new metaheuristic particle filter is proposed in this chapter. This algorithm is designed based on the combination of a generic particle filter and a modified tabu search optimisation algorithm.

The previous tracking trajectory used in most of the known filtering strategies (e.g. particle filter) is predominantly in the filtering process (i.e. the estimation of prior distribution for the next time step) in all prior research in this category. The novelty of this research is that the motion model and previous tracking trajectory are used in the metaheuristic optimization process to overcome the complexity induced by uncontrolled environments and targets. Using a previous trajectory (and motion model) in this way is not known to be applied in the related prior research.

The optimization process is constrained and supervised by the target's motion model. This strategy prevents the optimization process being dragged towards clutter in the background that may have similar visual features to those of the target of interest. Therefore, the new algorithm is more suitable than other algorithms for use in unconstrained environments with complex backgrounds.

The redundant evaluation processes, which are common in both generic particle filters and other metaheuristic optimization processes, are avoided by the implementation of

the tabu list. The goal of this strategy is to improve the performance of the overall system.

By imposing (1) diversification and intensification strategies through the tabu list, (2) the aspiration condition and (3) the classification of particles with different levels of fitness values, then this new algorithm will be able to thoroughly search the overall solution and at the same time converge to the known optimum quickly.

Compared with other algorithms in the literature, the new tabu search particle filter has a number of unique features.

Firstly, the optimisation process is constrained and supervised by the target's motion model. This strategy prevents the optimisation process being dragged towards clutters in the background that may have similar visual features to those of the target of interest. Therefore, the new algorithm is more suitable than other algorithms for use in unconstrained environments with complex backgrounds.

Secondly, the redundant evaluation processes, which are common in both generic particle filters and other metaheuristic optimisation processes, are avoided by the implementation of the tabu list. The goal of this strategy is to improve the performance of the overall system.

Finally, by imposing diversification and intensification strategies through the tabu list, the aspiration condition and the classification of particles with different levels of fitness values, this new algorithm will be able to thoroughly search the overall solution and at the same time converge to the known optimum quickly.

Chapter 6
The Colour Sequence Model

Colour histograms are usually used as the observation model in computer vision based tracking systems. They provide a robust, colour based, observation of the target, which is insensitive to scale change, rotation and partial occlusion.

The hue-saturation-value (HSV) colour space is commonly used for illumination invariant target tracking. The illumination invariant capability is provided by the HSV colour space because it separates the brightness from colour presentation. However, using the HSV colour space for illumination invariant measurement has a significant limitation: the system is unable to handle low saturation colours, which are very common in real world scenes, such as the presence of white, grey and black. Hue values in the HSV colour space are unreliable when the saturation channel is in the low range as is illustrated in figure 6.1 [67].

The motivation for the proposed colour sequence model (CSM) introduced in this chapter is to “shift” the responsibility of illumination invariance from the colour space to the observation representation model so that a reliable but luminance sensitive colour space (for example the RGB colour space) can be used in illumination invariant tracking systems.

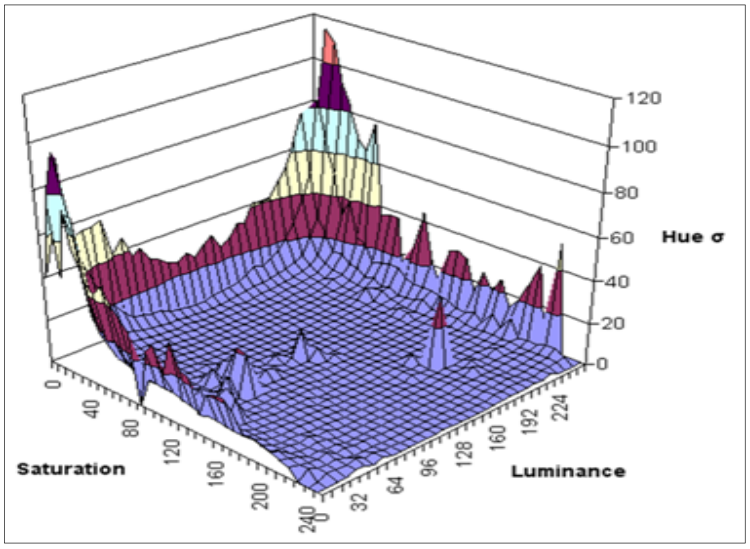


Figure 6.1 [67]: The noisy HSV colour space. As shown in this figure, the standard deviation of a pixel in HSV colour space is very large when the pixel's colour is in the low saturation area.

Designing illumination invariant models in the RGB colour space is not an easy task because all channels in the RGB colour space represent a mixture of both colour and luminance. Therefore, when illumination levels change in the environment, values of all channels are changed accordingly. A comparison of the RGB colour space and the HSV colour space given environmental illumination variations is illustrated in figure 6.2. As shown, all channels in the RGB colour space are changed as significantly as the value channel (luminance channel) in the HSV colour space whereas the hue channel in the HSV colour space is almost unchanged during the dramatic lighting strength variance in this test. The difficulty is to retain an object's identity given such dramatic change in RGB colour channels.

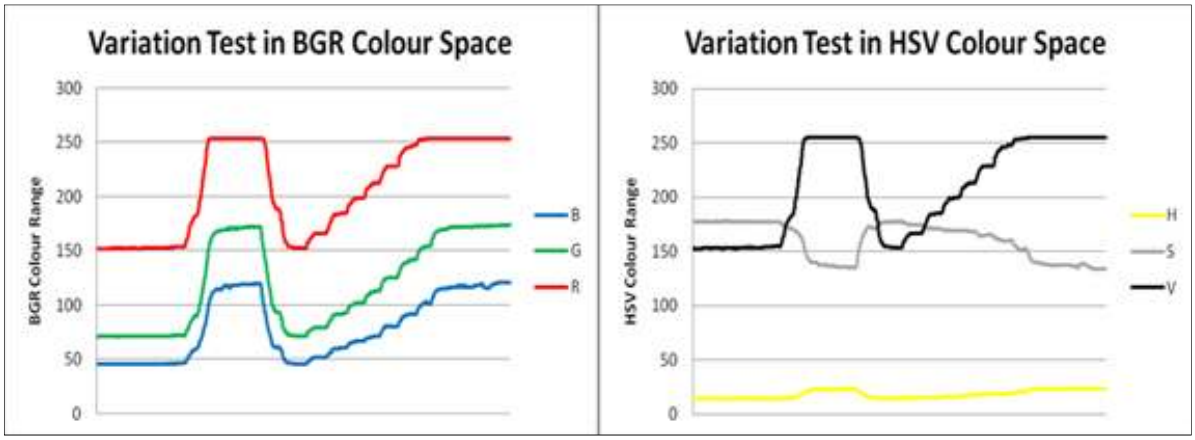


Figure 6.2: Compare RGB colour space and HSV colour space in various lighting conditions.

The proposed CSM was inspired by the continuous human movement recognition (CHMR) framework proposed in [68, 69]. In the CHMR model, activities of humans are deconstructed into an ordered collection of “dynemes”, which are defined as the smallest contrastive units of human motion. The dyneme alphabet was defined and collected by deconstructing hundreds of human movements into their lowest common denominator of basic movement patterns. A Hidden Markov model (HMM) was chosen in the CHMR framework to model human movement as an ordered sequence of dynemes selected from the alphabet. Human activity recognition was achieved by seeking the maximum likelihood of generating observed sequences from collected skill models through a HMM based algorithm.

In the proposed CSM, the image of an object is evenly segmented and indexed patches are obtained. These patches are defined as the smallest contrastive units of the object’s image, which is equivalent to the “dyneme” concept in [68, 69]. Through this method,

the colour properties of a target in a tracking system can be characterized as a sequence of non-stationary (moving and pose variation), corrupted (environment illumination variation) and continuous signals.

Compared with the widely used colour histogram model [70-79], the proposed CSM introduces a new dimension in colour based object matching criteria: namely, not only has the colour distribution of a region been taken into account, but the relative relationship between adjacent regions has also been taken into account through the first order Markov assumption. Therefore, it is more powerful than the colour histogram model in colour object recognition and colour matching tasks.

However, the one limitation is that although the CSM is motivated by using a HMM to process colour sequences, the proposed solution only uses a degenerated form of the HMM, as the transition probability is always 1.0. This is because in this chapter, the CSM is demonstrated in a “one object one model” manner. There is a one to one relationship between the object in a system and its corresponding CSM. Such limitations do not exist in systems where more than one object is represented by one CSM (for example, one CSM corresponds to multiple people).

This chapter is organized as follows: HMMs are reviewed in section 6.1. The proposed CSM is introduced in section 6.2 and 6.3.Then the CSM is evaluated in a rigid object comparison task in section 6.4.

6.1 Hidden Markov Models

In this section, key concepts of hidden Markov models are reviewed, especially those that are directly related to the discussion of the colour sequence model.

6.1.1 Definition of HMM

A typical HMM is illustrated in figure 6.3. The HMM shown in this diagram can be uniquely defined by five factors [68, 69, 80-84].

A HMM consists of a set of states, denoted as $S = \{S_1, S_2, S_3, \dots, S_N\}$, where N is the total number of states in the model.

At each state, the HMM emits a set of observable symbols, denoted as $V = \{V_1, V_2, V_3, \dots, V_M\}$, where M is the number of distinct observations.

The transition probabilities among all states in a HMM is defined by a $N \times N$ matrix, denoted as: $A = \{a_{ij}\}$ where $a_{ij} = P(q_{t+1} = S_j | q_t = S_i), 1 \leq i, j \leq N$.

The probabilities of emission for each state in a HMM is defined by an $N \times M$matrix, denoted as: $B = \{b_j(k)\}$ where $b_j(k) = P(o_t = V_k | q_t = S_j), 1 \leq j \leq N \; and \; 1 \leq k \leq M$.

The initial distribution of the HMM is coded in a vector of length N, denoted as $\pi = \{\pi_i\}$, where $\pi_i = P(q_1 = S_i)$.

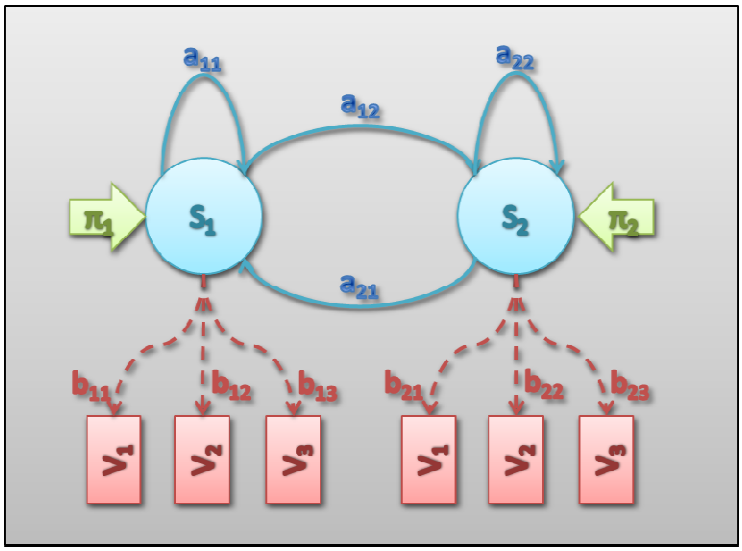


Figure 6.3: An example of a typical discrete HMM.

6.1.2 The Main Problems Addressed by HMMs

The HMM was developed to address some problems summarized in the following basic questions around a sequence of observable symbols [82, 84].

1. How well does a given HMM matched the observation?
2. How is the sequence of state transitions in the HMM defined such that the most likely observation is emitted?

3. How are the parameters of the given HMM modified so that the observation is more likely to be generated from this HMM?
4. If two or more HMMs are given, how is the similarity distance among the HMMs defined?

In this section, some common solutions are introduced from the literature, for example [68, 69, 80-84], with regards to these four commonly asked questions in HMM research.

Forward and Backward Algorithms

The forward [80, 84] and backward algorithms [80, 84] are two similar methods of measuring how well a given HMM matches a given observation. Both are recursive algorithms.

In the forward algorithm, a forward variable $\alpha_t(i)$ is defined as the probability that an observation sequence "$o_1 o_2 \dots o_t$" is observed up to time t and the state at time t is S_i. Namely:

$$\alpha_t(i) = P(o_1 o_2 \dots o_t, q_t = S_i|\lambda) \quad (6.2\text{-}1)$$

As suggested by its name, the forward algorithm processes the observation sequence in forward order. Therefore, the forward algorithm begins by defining the forward variable $\alpha_1(i)$, namely:

$$\alpha_1(i) = \pi_i b_i(o_1), 1 \leq i \leq N \quad (6.2\text{-}2)$$

Then, the process moves forward along the observation sequence, which is indicated by equation 6.2-3 as follows:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1}), 1 \leq j \leq N, 1 \leq t \leq T-1 \quad (6.2\text{-}3)$$

Finally, the forward algorithm stops at the end of observation sequence, namely:

$$\alpha_T(i) = P(o_1 o_2 \ldots o_T, q_T = S_i|\lambda) \quad (6.2\text{-}4)$$

The solution for question one can be found by calculating $P(O|\lambda)$ the sum of $\alpha_T(i)$ over all hidden states:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \quad (6.2\text{-}5)$$

In a similar manner, in the backward algorithm, a backward variable $\beta_t(i)$, which indicates the probability of the partial observation sequence from t + 1 to the end, given state S_i at time t, is defined as:

$$\beta_t(i) = P(o_{t+1} o_{t+2} \ldots o_T|q_t = S_i, \lambda) \quad (6.2\text{-}6)$$

The backward algorithm is a reverse process of the forward algorithm introduced above. Therefore, it begins from the end of the observation sequence, namely:

$$\beta_T(i) = 1, 1 \leq i \leq N \quad (6.2\text{-}7)$$

Then, the process moves backward along the observation sequence and is represented in equation 6.2-8 as:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), t = T-1, \ldots, 1; 1 \leq i \leq N \quad (6.2\text{-}8)$$

The backward process terminates at $\beta_1(i)$ and the solution of $P(O|\lambda)$ is given by:

$$P(O|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i) \quad (6.2\text{-}9)$$

Viterbi Algorithm

Developed in 1967 [85], the Viterbi algorithm computes the most likely hidden state sequence in a HMM, given an observation sequence.

There are two intermediate variables defined in the Viterbi process. The first variable, $\delta_t(i)$ is defined as the highest probability along a hidden state sequence ($Q = q_1 q_2 \ldots q_T$), which accounts for the first t observations and ends in state S_i. It can be represented as:

$$\delta_t(i) = \max_{q_1 q_2 \ldots q_t} P(q_1 q_2 \ldots q_t = S_i, o_1 o_2 \ldots o_t | \lambda) \quad (6.2\text{-}10)$$

The second variable $\psi_t(j)$ is defined to keep track of the argument which maximized $\delta_{t-1}(i)$, namely:

$$\psi_t(j) = \underset{1 \leq i \leq N}{\operatorname{argmax}} [\delta_{t-1}(i) a_{ij}] \quad (6.2\text{-}11)$$

The process of the Viterbi algorithm is summarized as follows. Firstly, the recursive Viterbi process begins with initializing both intermediate variables:

$$\delta_1(i) = \pi_i b_i(o_1), 1 \leq i \leq N$$
$$\psi_1(j) = null \quad (6.2\text{-}12)$$

Then, the intermediate variable $\delta_t(i)$ is recursively updated using:

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(o_t) \quad (6.2\text{-}13)$$

Accordingly, the intermediate variable $\psi_t(j)$ is updated with $\delta_t(j)$ based on its definition in equation 6.2-2. Finally, when all observations have been processed, the most likely hidden state sequence in a HMM that generates the observation sequence is given by:

$$q_t^* = \psi_{t+1}(q_{t+1}^*), t = T-1, T-2, \ldots 1 \quad 6.2\text{-}14$$

The corresponding probability of generating the given observation sequence is:

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)] \quad (6.2\text{-}15)$$

The Baum-Welch Algorithm

The method of adjusting the parameters of a HMM according to a given observation sequence is addressed by the Baum-Welch algorithm [82, 86].

There are two intermediate variables defined in the Baum-Welch algorithm. Firstly, the variable $\gamma_t(i)$ gives the probability of being in state S_i at time t, given the observation sequence: O. Namely:

$$\gamma_t(i) = P(q_t = S_i | O, \lambda) \quad (6.2\text{-}16)$$

Secondly, the variable $\xi_i(i, j)$ is defined as the probability of being in state S_i at time t and in state S_j at time t + 1, given the observation sequence: O. It can be represented as:

$$\xi_t(i,j) = P\left(q_t = S_i, q_{t+1} = S_j \middle| O, \lambda\right) \tag{6.2-17}$$

As implied by the above definitions in equation 6.2-16 and 6.2-17, the relationship of the two intermediate variables is given by:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) \tag{6.2-18}$$

Furthermore, the definitions of both Baum-Welch variables can be substituted by the forward variable and backward variable defined in equation 6.2-1 and equation 6.2-6 as:

$$\gamma_t(i) = P(q_t = S_i|O,\lambda) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)} \tag{6.2-19}$$

$$\begin{aligned}\xi_t(i,j) = P\left(q_t = S_i, q_{t+1} = S_j \middle| O, \lambda\right) &= \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{P(O|\lambda)} \\ &= \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}\end{aligned} \tag{6.2-20}$$

Based on the definitions introduced above, the expected number of transitions made from state S_i can be calculated by summing $\gamma_t(i)$ over time index t; where:

$$Expected\ number\ of\ transitions\ from\ S_i = \sum_{t=1}^{T}\gamma_t(i) \tag{6.2-21}$$

Similarly, the expected number of transitions from state S_i to state S_j is given by:

$$Expected\ number\ of\ transitions\ from\ S_i\ to\ S_j = \sum_{t=1}^{T-1}\xi_i(i,j) \tag{6.2-22}$$

The adjusted parameters of the given HMM, i.e. $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$, can be calculated from:

$$\begin{aligned} &\overline{\pi_i} \\ &= \text{expected number of times in state } S_i \text{ when } t \text{ equals to } 1 \\ &= \gamma_1(i) \end{aligned} \tag{6.2-23}$$

$$\begin{aligned} &\overline{a_{ij}} \\ &= \frac{\text{expected number of transitions from state } S_i \text{ to state } S_j}{\text{expected number of transitions from state } S_i} \\ &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \end{aligned} \tag{6.2-24}$$

$$\begin{aligned} &\overline{b_j(k)} \\ &= \frac{\text{expected number of times in state } S_j \text{ and observing symbol } V_k}{\text{expected number of times in state } S_j} \\ &= \frac{\sum_{t=1}^{T} \gamma_t(j), O_t = V_k}{\sum_{t=1}^{T} \gamma_t(j)} \end{aligned} \tag{6.2-25}$$

The parameter adjustment process can be performed iteratively until some predefined conditions are met, for example:

$$\left| P(O|\lambda) - P\left(O\middle|\bar{\lambda}\right) \right| \leq \varepsilon \tag{6.2-26}$$

The Distance between Two HMMs

Given two HMMs, for example: $\lambda_1 = (A_1, B_1, \pi_1)$ and $\lambda_2 = (A_2, B_2, \pi_2)$, the distance measure $D(\lambda_1, \lambda_2)$ is defined by the Kullback-Leibler divergence [87] as how well model λ_1 matches the observations generated by model λ_2, related to how the model λ_2 matches the observation generated by itself, that is:

$$D(\lambda_1, \lambda_2) = \frac{1}{T}[\ln(P(O^{(2)}|\lambda_1)) - \ln(P(O^{(2)}|\lambda_2))] \quad (6.2\text{-}27)$$

In this equation, $O^{(2)} = o_1 o_2 \ldots o_T$ is an observation sequence generated by the model λ_2.

The distance measurement defined above is not symmetric, i.e. $D(\lambda_1, \lambda_2) \neq D(\lambda_2, \lambda_1)$. A symmetrical version is usually defined as:

$$D_{sym}(\lambda_1, \lambda_2) = \frac{[D(\lambda_1, \lambda_2) + D(\lambda_2, \lambda_1)]}{2} \quad (6.2\text{-}28)$$

6.1.3 The Taxonomy of a HMM

HMMs are classified based on many different aspects [82, 84]. In this section two conditions related to the topology of a HMM or the emission properties of a HMM, that are most related to the discussion of the proposed CSM, are introduced.

Among all possible topologies of a HMM, the two most commonly used HMM topologies are the ergodic model and the Bakis model [88]. The ergodic model is also known as

the full connected model and in this model every hidden state can be reached from every other hidden state in a finite number of steps.

On the contrary, in a Bakis HMM, which is also known as a left-right HMM, the connections among hidden states are unidirectional, that is, the coefficients in its state transition matrix $A = \{a_{ij}\}$have the following properties:

$$a_{ij} = 0, for\ all\ j < i \tag{6.2-29}$$

Similarly, the coefficients in the initialization vector $\pi = \{\pi_i\}$ have the following properties:

$$\pi_i = \begin{cases} 0, & while\ i \neq 1 \\ 1, & while\ i = 1 \end{cases} \tag{6.2-30}$$

A HMM can also be classified based on the type of its observable symbols as either a discrete HMM or a continuous HMM.

A discrete HMM has a finite set of discrete observable symbols. For example, the observation alphabet in the HMM shown in figure 4.2.1-1 has three elements, namely $V_1, V_2, and\ V_3$. As mentioned earlier, the emission probability of a discrete HMM can be described by a matrix, denoted as: $B = \{b_j(k)\}$ where $b_j(k) = P(o_t = V_k | q_t = S_j), 1 \leq j \leq N\ and\ 1 \leq k \leq M$.

On the other hand, in a continuous HMM, the emission probability is a continuous probability density function (*p.d.f*) and usually approximated by a sum of M weighted distributions [82, 84] and represented in general as:

$$b_j(O) = \sum_{m=1}^{M} c_{jm} \mathfrak{N}[O, \mu_{jm}, U_{jm}], \qquad 1 \leq j \leq N \tag{6.2-31}$$

In equation 6.2-31, O is the observation vector being modelled, and c_{jm} the mixture coefficient of the m-th mixture in state S_j. $\mathfrak{N}$ is any logarithmically concave [89] or elliptically symmetric [86] probability density function with mean vector μ_{jm} and covariance matrix U_{jm} for the m-th mixture component in state S_j.

6.2 The Colour Sequence Model

The CSM developed in this research is a unique method of quantitatively measuring the similarity of two coloured objects. It can be applied in the applications that require matching multiple objects based on their appearances, for example a human tracking system.

6.2.1 The Definition of the New CSM

The new CSM is based on a continuous Bakis HMM with a trivariate Gaussian emission distribution. The basic structure of the CSM is illustrated in figure 6.4.

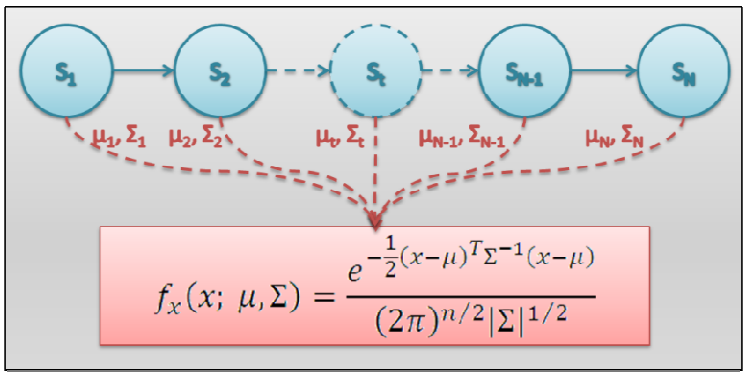


Figure 6.4: The HMM structure underneath the CSM.

The sample CSM illustrated in the figure contains N hidden states in total, i.e. $S = \{S_1, S_2, S_3, \dots, S_N\}$. The allowed state transitions are drawn as blue coloured arrows in the model and as shown, only state transition from one state to its immediate succeeding state are allowed, such that:

$$transition\ matrix\ A = \{a_{ij}\}$$
$$a_{ij} = \begin{cases} 0, & while\ j \neq i+1 \\ 1, & while\ j = i+1 \end{cases} \quad (6.3\text{-}1)$$

Furthermore, the observation *p.d.f* of the proposed CMS is defined by a trivariate Gaussian distribution:

$$b_j(O) = f(x;\ \mu,\Sigma) = \frac{\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)}{(2\pi)^{n/2}|\Sigma|^{1/2}} \quad (6.3\text{-}2)$$

In the above equation, the variable vector x and mean vector μ contain three components respectively and the covariance matrix Σ is a three by three matrix, which are given by:

$$x = \begin{pmatrix} x_{c1} \\ x_{c2} \\ x_{c3} \end{pmatrix}, \qquad \mu = \begin{pmatrix} \mu_{c1} \\ \mu_{c2} \\ \mu_{c3} \end{pmatrix}$$

$$\Sigma = \begin{bmatrix} \rho_{11} & \rho_{12} & \rho_{13} \\ \rho_{21} & \rho_{22} & \rho_{23} \\ \rho_{31} & \rho_{32} & \rho_{33} \end{bmatrix} = \begin{bmatrix} \sigma_{c1}^2 & 0.0 & 0.0 \\ 0.0 & \sigma_{c2}^2 & 0.0 \\ 0.0 & 0.0 & \sigma_{c3}^2 \end{bmatrix} \tag{6.3-3}$$

6.2.2 Life Cycle of the New CSM

The Initialization Step of the CSM

The Baum-Welch algorithm discussed in section 6.2 is simplified in the proposed CSM because it has a simple topology structure and observation *p.d.f.*. As shown in figure 6.4, in the CSM, the transition probability a_{ij} is either equal to one if S_j is the succeeding state of S_i, i.e. $j = i + 1$, or, otherwise, zero. Therefore, the expected number of transitions from S_i *to* S_j is either equal to the expect number of transitions from S_i, if S_j is the succeeding state of S_i or zero in other cases. Furthermore, the observation *p.d.f.* of the CSM indicates that the colour signals in each channel are modelled as standalone Gaussian distribution with mean μ_c and variance σ_c^2, where:

$$b_j(O_c) = f(x_c) = \frac{\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}{\sigma_c\sqrt{2\pi}} \tag{6.3-4}$$

The initialization process of the proposed CSM is illustrated in figure 6.5 and is described as follows.

Firstly, the latest input image is converted to an observation sequence (using algorithms that will be introduced later). The converted observation contains a sequence of N observable values, i.e. a sequence of average colour values of each image patch.

Then, the basic structure of the CSM is established as a Bakis HMM with prior known state transition matrix coefficients and some initial parameters for the observation *p.d.f.*

Once the basic HMM structure is known, the parameters of the initial observation *p.d.f.* are adjusted iteratively according to the observation sequence converted earlier, so that the available observation sequence is most likely to be generated by an established CSM.

Finally, all parameters, including state transition matrix of the HMM and the parameters of the observation *p.d.f.* of each hidden state are verified by rules of probability [90] and stored as the initial parameters of the CSM. The variance of the Gaussian based emission distribution at each channel is specified manually.

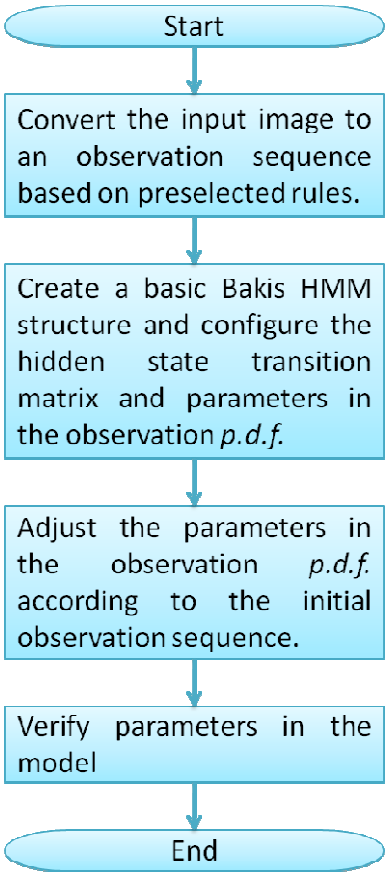


Figure 6.5: The algorithm for initializing the CSM.

The Comparison Step of the CSM

As mentioned earlier, the CSM was developed as an effective algorithm in an object tracking system, especially for disambiguating multiple targets in a small area, when their identity is unable to be uniquely identified based on their positions. This new colour model offers this functionality through its similarity comparison step.

The similarity between two colour images in the context of the CSM is defined as the distance between the two HMMs based on the images. Calculating the distance between two CSMs is simplified as the probability of the second observation being generated from the first CSM is given by:

$$Similarity\ in\ CSM = P\left(O^{(2)}\middle|\lambda_1\right) \tag{6.3-5}$$

The above equation can be solved by either forward algorithm or backward algorithm.

The algorithm for calculating the distance between two colour objects is illustrated in figure 6.6. As shown in this flow chart, the new colour image is converted to an observation sequence. Then, the likelihood of this observation sequence being generated by the existing CSM is calculated through a forward algorithm. If there are more than one pair of colour objects being compared, all these comparisons are calculated and their likelihood values are normalized so that they can be compared with each other.

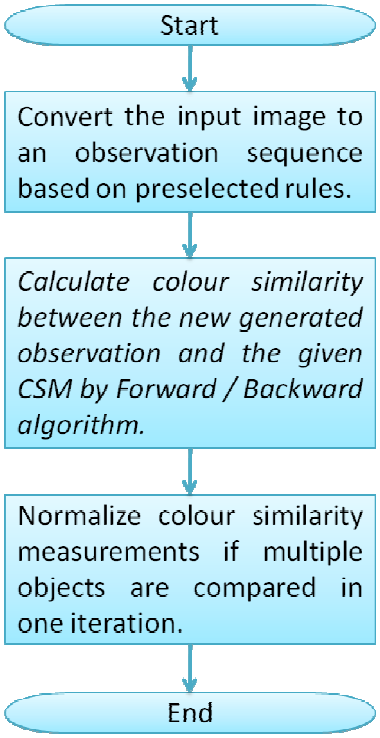


Figure 6.6: The algorithm for comparing the distance between two colour objects in the context of CSM.

The Updating Step of the CSM

The colour of an object in a tracking system may change overtime, such as a result of the target's movement or changing environmental illumination.

The updating process of the CSM can be simplified to a process of continuously adjusting parameters for the Gaussian observation *p.d.f.* defined in a CSM, which can be achieved through a running average algorithm with a predefined update rate: α_t. The

observation and state variables are associated with the cell signals and Gaussian parameters. This process can be represented in the following equation:

$$\begin{cases} \mu_t^{S_i} = \mu_{t-1}^{S_i} * \alpha_t + \begin{pmatrix} O_t^{c1} \\ O_t^{c2} \\ O_t^{c3} \end{pmatrix} * (1 - \alpha_t); \\ \alpha_t \propto 1.0 - P(O_t|\lambda_{t-1}) \end{cases} \quad (6.3\text{-}6)$$

The CSM updating algorithm is illustrated in figure 6.7. As shown, the parameters of the current CSM are adjusted according to the latest observation sequence. The speed of the updating process is controlled by a predefined factor, named CSM updating ratio, which is in the range from zero to one. As shown in the flow chart below, the updated parameters are verified by probabilistic rules before they are confirmed and saved as the new CSM.

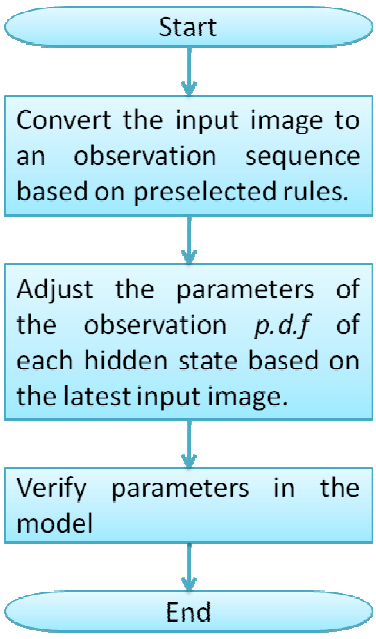


Figure 6.7: The algorithm for updating a CSM with a new input image.

6.2.3 Converting Colour Images to Observation Sequences

The image-to-sequence conversion algorithm developed involves two steps: a decomposing step and a linking step.

In the decomposing step, an image with a coloured object is evenly segmented by horizontal and vertical lines and indexed patches were obtained. These indexed patches are the smallest contrastive units of the appearance of a coloured object. Figure 6.8 illustrates the decomposing step with an example image of a person walking. As shown, the original image (leftmost in the figure) is segmented by six evenly distributed vertical and horizontal lines. The whole image becomes sixteen patches as shown in the middle image in the figure. These patches are then indexed with their horizontal and vertical positions as shown in the rightmost image in the figure and will be chained through some methods in the linking step to form a sequence of observations of the input image.

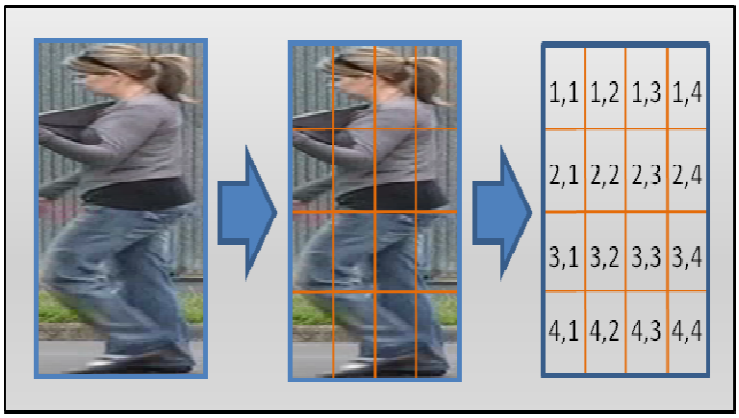


Figure 6.8: The decomposing step in the proposed algorithm.

In the linking step, a first order Markov assumption is imposed between any adjacent image patches. Because the image patches are decomposed from the original image, there are no intrinsic meaningful connections among these patches. Therefore, there is no natural method of linking the segmented image into a sequence of events. The methods used in the linking step are chosen based on the application properties. In the rest of this section, three different methods of linking indexed image patches are introduced. The optimal areas of application for each method will also be introduced.

The first linking method is illustrated in figure 6.9. As shown, the patches at the same row are grouped together as one observation event. The whole image contains four such observation events, which are marked as 1, 2, 3 and 4. These observations are then linked up to a sequence of length four. Three dimensional vectors of the mean value of each observation event are listed in the table shown at the top-right region of

the figure. The bottom part of figure 6.9 illustrates the intensity variation of the four observations in RGB colour space.

The method introduced above is useful in human tracking applications for the following reasons: (1) Horizontal ego-movements usually exist in human walking action. For example, a walking person may swing his/her arms and the legs are moving forward and backward alternately relative to the torso. The method used here groups horizontal patches together as one observation event and therefore, the appearance variations caused by the ego-movements mentioned above are eliminated or minimized. (2) As the value of each observation event is calculated from the average of a combined area of four patches, the changes of appearance, as a result of different angles of observation or different background textures, between adjacent frames are minimized. Therefore, the observations are more stable than some linking methods with small patches. (3) As the sequence generated with this method is relatively short. The time used for initialization, matching and update CSMs is minimized. Therefore, it is suitable for time-critical applications, such as human tracking.

Figure 6.9: The indexed image patches are linked together as a vertical chain.

The other two examples of a linking method are similar to each other and they are illustrated in figure 6.10 and figure 6.11 respectively. As shown, in both methods, the sixteen patches are grouped into four square macro patches. The macro patches are chained either in a zigzag sequence or in a ‘snake’ sequence.

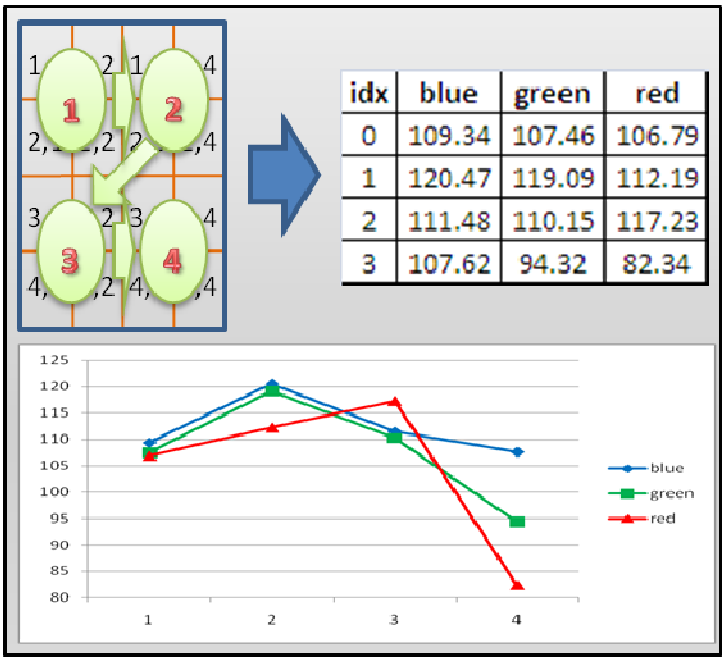


Figure 6.10: The indexed image patches are linked to a sequence of observation in a zigzag sequence.

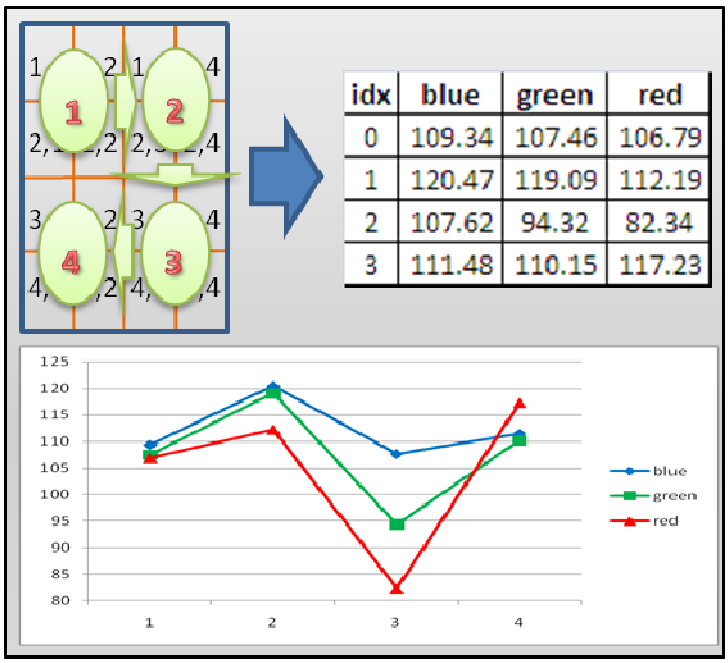
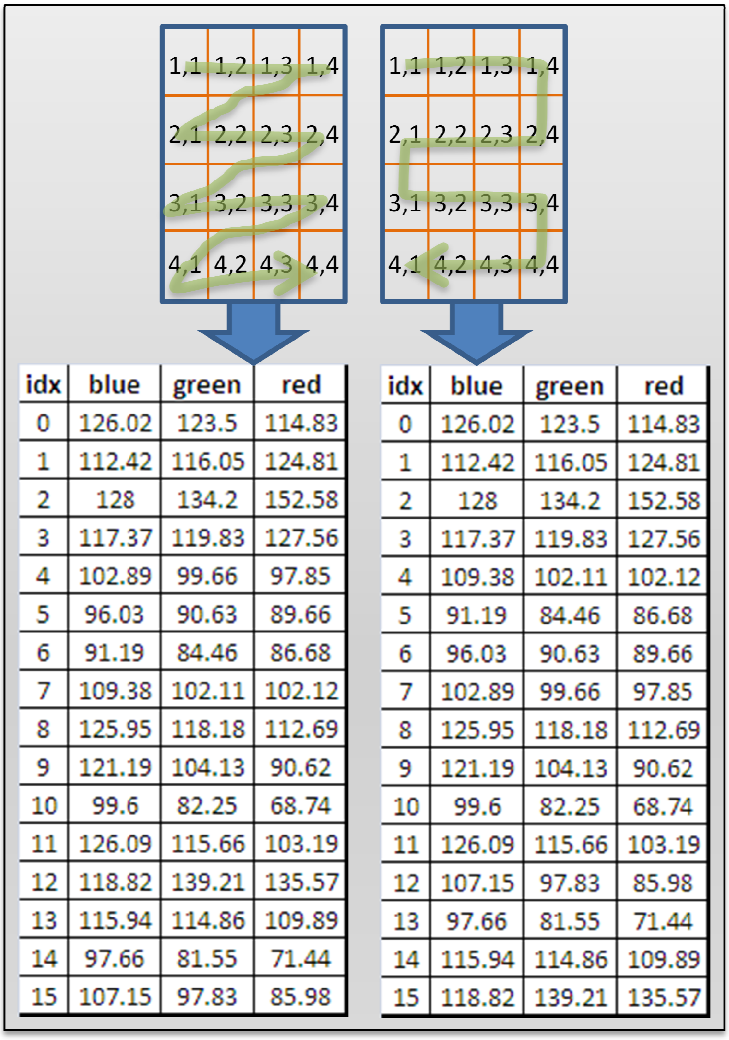


Figure 6.11: The indexed image patches are linked to a sequence of observation in a 'snake' sequence.

Comparing these latter two methods (zigzag and 'snake') with the vertical method already discussed, a notable difference is that through these two methods long observations can easily be generated. For example, figure 6.12 shows that two sequences of observations with sixteen in length are generated through the zigzag and 'snake' methods. The advantages of having long observation sequences is that the imposed Markov assumptions are stronger than for short sequences because there are more observations to test whether the criteria is satisfied. Therefore, the test conditions on each single observation in the sequence can be relaxed to allow a wider tolerance to appearance variations caused by, for example, changing illumination. On the other hand, the disadvantage of long sequence is that it is more sensitive to the deformation of non-rigid objects, such as a moving human body, than a short sequence. This is because given the same number of patches, a single observation event along a long sequence is generated from a much smaller area of the image compared with an observation event with short sequences. Therefore, the zigzag method and 'snake' method are more preferable in applications that match or compare rigid objects with large variation in lighting conditions.



idx	blue	green	red
0	126.02	123.5	114.83
1	112.42	116.05	124.81
2	128	134.2	152.58
3	117.37	119.83	127.56
4	102.89	99.66	97.85
5	96.03	90.63	89.66
6	91.19	84.46	86.68
7	109.38	102.11	102.12
8	125.95	118.18	112.69
9	121.19	104.13	90.62
10	99.6	82.25	68.74
11	126.09	115.66	103.19
12	118.82	139.21	135.57
13	115.94	114.86	109.89
14	97.66	81.55	71.44
15	107.15	97.83	85.98

idx	blue	green	red
0	126.02	123.5	114.83
1	112.42	116.05	124.81
2	128	134.2	152.58
3	117.37	119.83	127.56
4	109.38	102.11	102.12
5	91.19	84.46	86.68
6	96.03	90.63	89.66
7	102.89	99.66	97.85
8	125.95	118.18	112.69
9	121.19	104.13	90.62
10	99.6	82.25	68.74
11	126.09	115.66	103.19
12	107.15	97.83	85.98
13	97.66	81.55	71.44
14	115.94	114.86	109.89
15	118.82	139.21	135.57

Figure 6.12: Long observation sequences can be generated through zigzag or 'snake' methods.

6.3 Quantitative Comparison of the CSM and CHM

6.3.1 Test Data and Experimental Framework

Figure 6.13 shows four examples of images used in this experiment. In this figure, the two images located in the left column were captured in a sunny outdoor environment. The very light areas in the right-bottom corners of these two images were sunlight

reflections. The two images on the right side of the figure were captured in a darker indoor environment. These four images were captured from the <Canterbury Bluebook> issue 509. The two images at the first row are page 27 of this book and the bottom two images are page 47 of this book. The whole book contains 84 pages and 80 pages were selected to be used in this rigid object matching experiment.

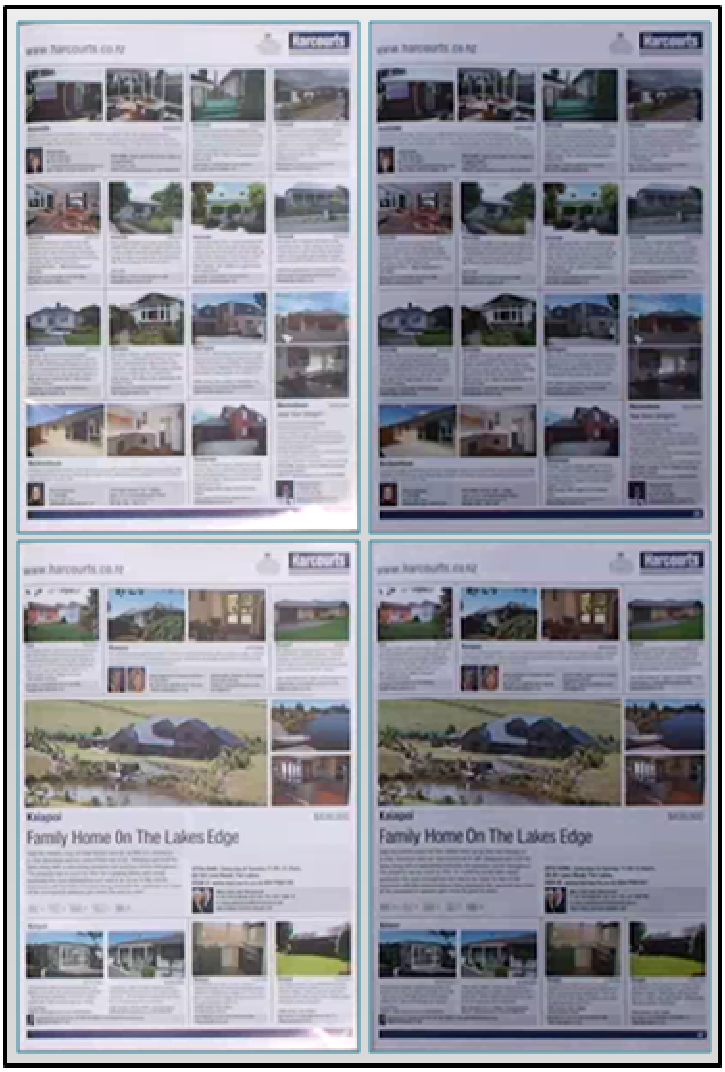


Figure 6.13: Examples of images used in this experiment.

The images of the pages from the book used in the experiments were extracted from two video clips of the test application. Once the four corners of a page are known, a perspective transformation was performed to correct the perspective distortion of the book page and the page is normalized to a 240 by 320 pixel rectangular image. All images captured from the same video clip were stored in one folder and an xml control file was generated to manage these image files. An example of such folder structure and the content of the control file are illustrated in figure 6.14. In the rigid object matching experiment discussed in this section, a total of 160 images captured from two video clips were used. These images were classified into the two categories, sunny-outdoor group and dark-indoor group. Each group contained 80 images.

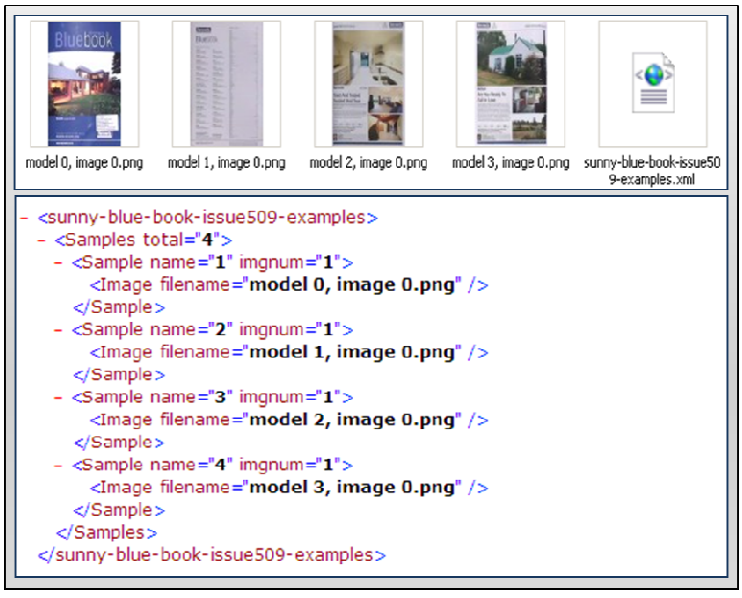


Figure 6.14: Examples of the folder structure and xml control file of a collection with four images.

The experimental process was as follows: the test application parsed the xml control file in the folder (training set) and generated CSM and CHM for each image in that folder. Then, the test images in the comparing folder (test set) were loaded and matched with the trained models. The matching results were logged including both the accuracy test results and the performance test results.

6.3.2 Analysis of the Accuracy and Performance of CSM

The accuracy and performance of the proposed CSM was measured with the rigid object matching algorithm introduced in the previous section.

The goal of this experiment was to determine the degree to which configurations of the CSM, such as the variances of observation functions and the size of image patches, influence accuracy and performance of CSM. As introduced previously, the resolution of a normalized image used in this experiment was 240 pixels in width and 320 pixels in height. The resolution of image patches was indicated by the number of image patches in both horizontal and vertical directions. For example, in figure 6.15, the item '04by08' indicates that there were four horizontal image patches and eight vertical image patches that have been segmented, where, each image patch in this case is 60 pixels in width and 40 pixels in height. The 'var' values in the following figures in this section indicate the variance value of the observation functions. The accuracy in these experiments was measured by correct matching rate (CMR) value. This is the percentage of correctly matched pages among the total number of images used to train models. The

performance was measured by the average time (in milliseconds) taken to compare one image against one model.

The following three figures (figure 6.15, figure 6.16 and figure 6.17) detail the accuracy results of the proposed CSM under different parameter configurations. The first figure (figure 6.15) shows accuracy with ten image models. In this test, ten images were captured in a dark indoor environment so that a set of ten CSMs could be trained. Then, another set of images of the same ten pages were captured under sunny outdoor lighting conditions. These brighter images were used for comparison with the darker CSMs. The results of this test are shown in a table and graph. As shown in the graph, the CSM achieved a 100% CMR when the variances of observation functions are larger than sixteen and the image patch size is smaller than 40 pixels by 27 pixels (06by12). In addition, it is clear in this graph that when the variances are small (for example 2), a larger patch size configuration (which means coarser resolution and shorter observation sequence) improves the matching rate of the matching process. This is because large patch size (coarser resolution) eliminates the effects of irregular local illumination changes, such as the high reflection areas shown in figure 6.13.

	var=2	var=4	var=6	var=8	var=16	var=32	var=64	var=128	var=256
04by08	70%	90%	90%	90%	90%	90%	90%	90%	90%
06by12	60%	70%	80%	80%	100%	100%	100%	100%	100%
08by16	60%	70%	70%	80%	100%	100%	100%	100%	100%
10by20	50%	70%	70%	80%	100%	100%	100%	100%	100%

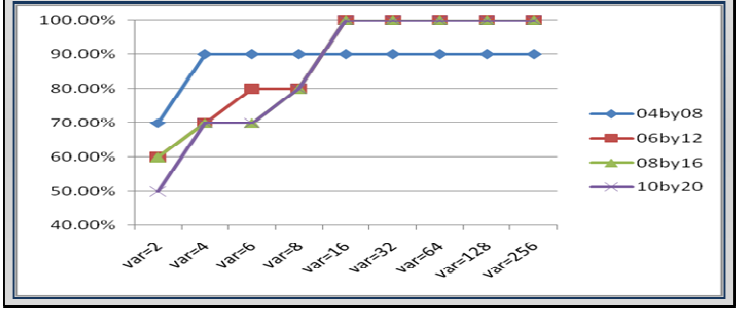


Figure 6.15: The accuracy of the CSM with ten images under different patch scales and variances of observation functions.

Figure 6.16 and figure 6.17 illustrate the test results of the same experiments as discussed above. However, these two tests were carried out with image samples of 40 images and 80 images respectively. As shown, compared with the test results shown in figure 6.15, the correct matching rates in both experiments decreased slightly when the number of models is increased. In the test with 40 images, the maximum CMR dropped to 90% from 100% in the previous test and it dropped another 4% when the image samples were doubled (80 images). As we will see later in the next section, this CMR accuracy was much higher than the widely used CHM under the same test conditions.

	var=2	var=4	var=6	var=8	var=16	var=32	var=64	var=128	var=256
04by08	63%	68%	68%	68%	68%	68%	68%	68%	68%
06by12	55%	65%	73%	73%	75%	75%	75%	75%	75%
08by16	50%	70%	73%	78%	85%	85%	85%	85%	85%
10by20	45%	78%	83%	80%	90%	90%	90%	90%	90%

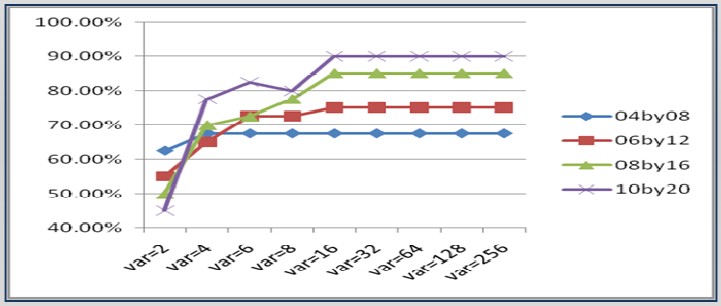


Figure 6.16: The accuracy of the CSM with 40 images under different patch scales and variances of observation functions.

	var=2	var=4	var=6	var=8	var=16	var=32	var=64	var=128	var=256
04by08	55%	59%	59%	59%	59%	59%	59%	59%	59%
06by12	58%	63%	66%	66%	68%	68%	68%	68%	68%
08by16	50%	66%	75%	79%	83%	83%	83%	83%	83%
10by20	45%	74%	78%	83%	89%	86%	86%	86%	86%

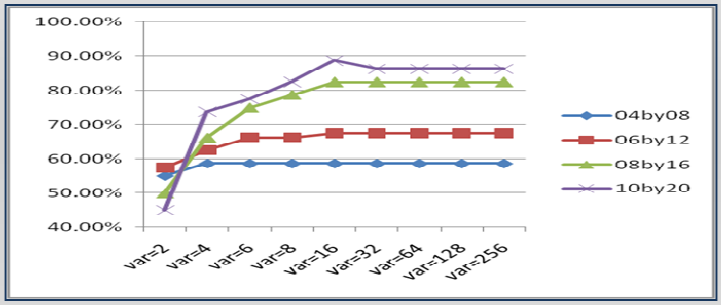


Figure 6.17: The accuracy of the CSM with 80 images under different patch scales and variances of observation functions.

The experimental results of all tests are illustrated in figure 6.18 and figure 6.19 with a selected image patch size (10by20, that is, 24 pixels by 18 pixels in each patch) and a selected Gaussian variance (sixteen). The highest resolution used in these tests

(10by20) output the best overall accuracy rates and the variance value sixteen generated the best CMR for any given resolution.

Figure 6.18 shows the test results of all eight experiments (with 10 images, 20 images, 30 images, 40 images, 50 images, 60 images, 70 images and 80 images) under nine different variances of observation functions. As shown, although the correct matching rates decrease slightly while the total number of models (image) to match is increased, the degree to which the matching speed falls is minimal with the values remaining at comparatively steady levels. For example, when the sample size was increased by eight times from 10 images to 80 images, the CMR only dropped 11.25% for a variance of sixteen.

Another interesting finding is illustrated in Figure 6.18 in the red rectangle: When the number of samples was increased from 10 images to 40 images, the recognition accuracy was increased (rather than decreased). Also shown was that the three curves go against the trend in this area representing middle range variances, i.e. 4, 6, and 8. By analyzing the image samples from both videos, the reason for this unexpected increasing CMR is found to be caused by a high level of local reflections in several images between sample one and sample 30. Given the fine resolution of observation sequences generated (small patch size), the variances values used here (4, 6 and 8) were too small to deal with such irregular local illumination conditions. As shown, such an effect does not appear in the curves representing larger variance values (such as

sixteen or above), because a CSM with larger observation variance values is less sensitive to such irregular local illumination changes. Also shown is that this effect does not appear in the graphed results with very small variance parameters such as a variance value of 2, because in this case, the CMR is too low (about 50% to 43%) to demonstrate negative consequences of irregular high illumination patches.

	10 images	20 images	30 images	40 images	50 images	60 images	70 images	80 images
var = 2	50.00%	50.00%	43.33%	45.00%	44.00%	45.00%	45.71%	45.00%
var = 4	70.00%	65.00%	73.33%	77.50%	74.00%	75.00%	75.71%	73.75%
var = 6	70.00%	70.00%	80.00%	82.50%	80.00%	76.67%	77.14%	77.50%
var = 8	80.00%	75.00%	73.33%	80.00%	80.00%	78.33%	81.43%	82.50%
var = 16	100.00%	100.00%	90.00%	90.00%	88.00%	86.67%	87.14%	88.75%
var = 32	100.00%	100.00%	90.00%	90.00%	86.00%	85.00%	85.71%	86.25%
var = 64	100.00%	100.00%	90.00%	90.00%	86.00%	85.00%	85.71%	86.25%
var = 128	100.00%	100.00%	90.00%	90.00%	86.00%	85.00%	85.71%	86.25%
var = 256	100.00%	100.00%	90.00%	90.00%	86.00%	85.00%	85.71%	86.25%

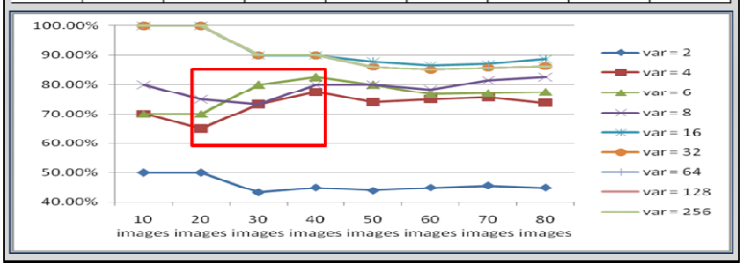


Figure 6.18: The accuracy of the CSM with patch scale at 10by20 (24 pixels by 18 pixels) and different variances of observation functions.

Figure 6.19 also shows the test results for all eight experiments under four different sizes of image patches (the minimum unit of constructing a sequence of observations). The variance of the Gaussian observation functions specified in this figure is sixteen. As shown, the observation sequences with higher resolutions performed better in terms of CMR than the observation sequences with lower resolutions.

	10 images	20 images	30 images	40 images	50 images	60 images	70 images	80 images
04by08	0.9000	0.8500	0.7000	0.6750	0.6200	0.6000	0.5857	0.5875
06by12	1.0000	1.0000	0.8000	0.7500	0.7200	0.6833	0.6857	0.6750
08by16	1.0000	1.0000	0.8667	0.8500	0.8200	0.8167	0.8286	0.8250
10by20	1.0000	1.0000	0.9000	0.9000	0.8800	0.8667	0.8714	0.8875

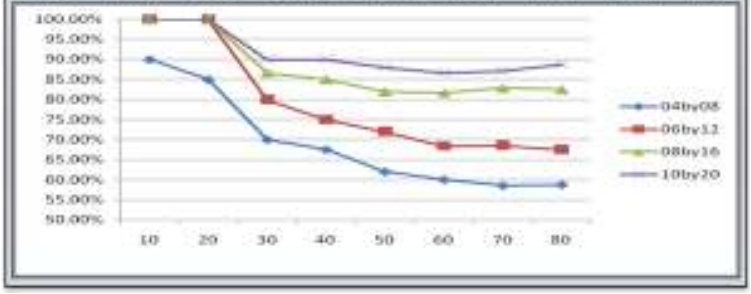


Figure 6.19: The accuracy of the CSM for a variance of observation functions of sixteen and for different patch scales.

Finally, figure 6.19 illustrates the performance test results as the time used (milliseconds) for comparing an image against a generated CSM. This performance measure is of the time taken to convert an image to a sequence of observation events and does not include the initial time taken to generate the CSMs. As expected, the time consumed by matching images against a high resolution CSM was longer than a low resolution CSM because of more calculations needed for evaluating the likelihood of generating a sequence from a longer HMM than a shorter one. An interesting finding illustrated in this figure is that the variances of Gaussian observation functions also significantly affects the computational efficiency as a consequence of the “cut-off threshold” used in the forward algorithm. Once the likelihood of an observation, as generated by a hidden state, was less than this threshold, the calculation was stopped. By this approach, images that were not generated from a given CSM could be identified quickly without processing the whole sequence of observations. However, a large

variance in the observation function decreases the probability that an impossible sequence was rejected at early stage and so more calculations were unnecessarily made to perform the comparison.

	var=2	var=4	var=6	var=8	var=16	var=32	var=64	var=128	var=256
04by08	0.52	0.56	0.58	0.59	0.62	0.61	0.60	0.60	0.59
06by12	1.17	1.03	1.08	1.17	1.49	1.68	1.64	1.59	1.57
08by16	1.45	1.90	2.32	2.54	3.21	3.98	4.17	4.13	4.09
10by20	3.17	3.97	4.76	5.91	7.50	10.45	11.56	11.15	11.18

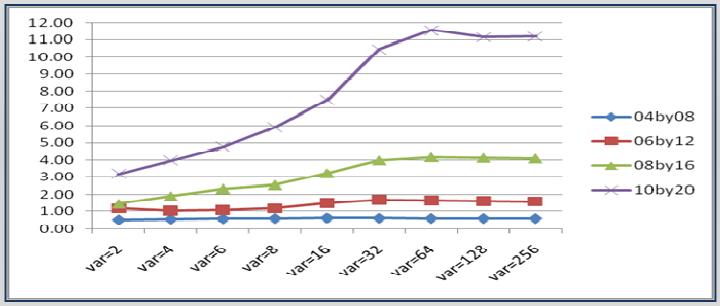


Figure 6.19: The performance test results of rigid object matching experiments.

6.3.3 Comparing the Accuracy of CSM and CHM

As discussed earlier, the CHM is a sophisticated coloured object matching algorithm widely used in computer vision based tracking applications, especially human tracking systems. In this section, the proposed CSM is compared with the CHM for accuracy of matching rigid objects based on colour.

Figure 6.20 illustrates the CMR values of a standard CHM tested in this research. As shown, although CMR values reached as high as 70% when the size of image database (the number of colour histograms being trained) was limited to less than 20 images, the CMR values dropped dramatically when the number of images was increased. For example, the best average CMR values were found in figure 6.16 with a colour histogram resolution of the six, where CMR values dropped from 70% for only ten images to 14.75% when the number of images increased to 80. The term resolution used in the context of CHM means the number of bins used to model the colour values of each channel in the RGB colour space.

	res=2	res=4	res=6	res=8	res=10	res=12	res=14	res=16	res=18
10 images	20.00%	70.00%	70.00%	70.00%	60.00%	60.00%	60.00%	50.00%	50.00%
20 images	10.00%	55.00%	70.00%	60.00%	55.00%	50.00%	55.00%	50.00%	45.00%
30 images	6.67%	40.00%	56.67%	56.67%	50.00%	43.33%	46.67%	46.67%	43.33%
40 images	5.00%	35.00%	35.00%	37.50%	30.00%	32.50%	30.00%	35.00%	30.00%
50 images	2.00%	24.00%	24.00%	18.00%	28.00%	20.00%	22.00%	24.00%	18.00%
60 images	1.67%	20.00%	20.00%	15.00%	21.67%	15.00%	13.33%	15.00%	11.67%
70 images	1.43%	21.43%	15.71%	18.57%	15.71%	12.86%	11.43%	12.86%	10.00%
80 images	1.22%	16.00%	14.75%	13.75%	15.00%	9.75%	8.50%	10.75%	7.25%

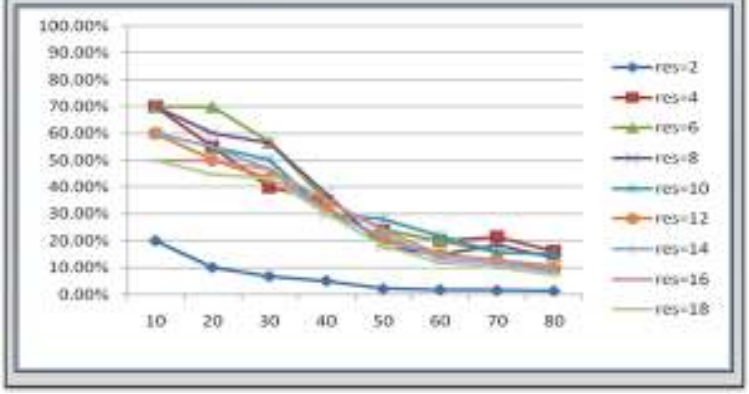


Figure 6.20: Accuracy test results of the standard CHM at different resolutions.

Figure 6.21 illustrates the differences of the CMR values between CSM and CHM over eight experiments. The accuracy test results generated by the CHM with a resolution of six (six bins are used to model colour values on each channel) was chosen to be used in this comparison because according to figure 6.16, the colour histogram at this resolution level outputs the best average accuracy in the experiments. As shown in the figure, two CMR results were selected from CSM accuracy results. The 'CSM-best' result (plotted in the graph with a red curve) was the CSM with a 10by20 resolution and a variance equal to sixteen. According to the experiments discussed in the previous section, these configurations output the best CMR value among all CSMs. However, the processing time of this configuration was 7.50ms per image, which is much higher than the CHM at 1.99ms per image. Therefore, another CSM configuration was also chosen in this comparison graph. The green curve in the figure plots the accuracy result of a CSM with a 6by12 resolution and a variance of sixteen. The processing speed of this CSM is 1.49ms per image, which is about three quarters the processing time of the colour histogram and the accuracy rate was double (2.09 times) that of the CHM.

	10 images	20 images	30 images	40 images	50 images	60 images	70 images	80 images
CHM-best	70.00%	70.00%	56.67%	35.00%	24.00%	20.00%	15.71%	14.75%
CSM-best	100.00%	100.00%	90.00%	90.00%	88.00%	86.67%	87.14%	88.75%
CSM-faster	100.00%	100.00%	80.00%	75.00%	72.00%	68.33%	68.57%	67.50%

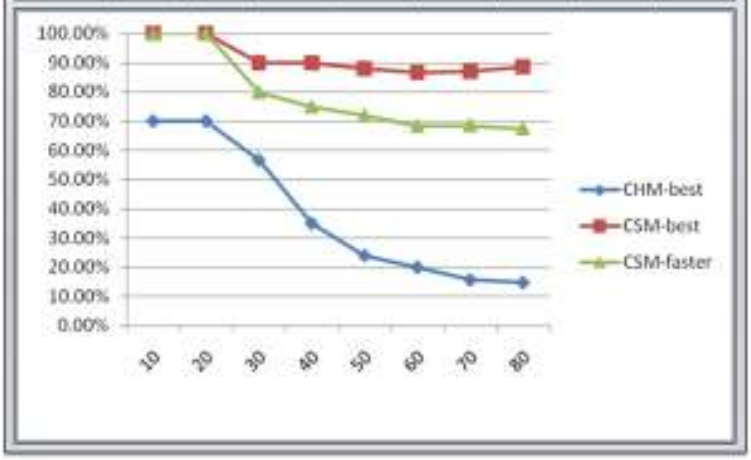


Figure 6.21: Compare of the accuracy test results between CSM and CHM.

6.4 Summary

In this chapter, a novel algorithm in object tracking domain, namely the CSM, is described. The motivation of this research was to find an efficient algorithm that was able to match a colour object under changing illumination conditions in RGB colour space so that some shortcomings of using HSV colour space could be eliminated.

The proposed CSM achieves this research motivation by introducing a new dimension in colour object comparison, being the imposed first order Markov assumption. The HMM was chosen as the fundamental mathematic basis of the proposed CSM to quantify the similarities of multiple CSMs.

The proposed CSM has been quantitatively evaluated with a rigid object matching experiment. The experimental results showed that the CSM is a significantly improvement over the colour histogram in both matching accuracy and speed.

Chapter 7
Results

In this chapter, the proposed tabu search particle filter is evaluated quantitatively in a real-time human tracking system with video footage taken from unconstrained outdoor environments.

In order to evaluate and compare the new tabu search particle filter to existing algorithms, the metaheuristic particle filters introduced in chapter 4 have been implemented and integrated into a human tracking system framework. The same human tracking system framework is also used for the tabu search particle filter. Therefore, all algorithms share the same program framework and particle filtering process (including prediction, observation and resampling).

This chapter is organized as follows: The experiment environment (apparatus) will be introduced in section 7.1. This is followed by a discussion of the video footage used in the experiments in section 7.2. The experiment methods are discussed in detail in section 7.3. In sections 7.4 the methods commonly used to retrieve the ground truth for computer vision based tracking systems are introduced. In section 7.5 the performance matrix of the new tabu search particle

filter will be illustrated quantitatively and compared with other existing metaheuristic particle filters.

7.1 The Experiment Environment

All experiments discussed in this chapter were performed using computers with the hardware and software profiles summarized in the following table.

Hardware Profile	
CPU	Intel Core 2 Quad Q6600 @ 2.40GHz
RAM	4.0GB Dual-Channel DDR2 @ 399MHz
Motherboard	Intel Corporation DQ35JO
Graphics	Philips 190S @ 1280x1024 512MB GeForce 9500 GT
Hard Drives	244GB Western Digital WDC WD2500AAKS-00B3A0

Software Profile	
Operating System	MS Windows XP Professional 32-bit SP3
OpenCV	Version 2.10
C++ Compiler	Microsoft Visual Studio 2008 Academic Edition

Table 7.1: A list of hardware and software profiles of the computers used in the experiments.

7.2 Video Footage

In this research, all experiments are performed using pre-recorded video footage to ensure that the test conditions remained the same for all algorithms being compared. The video footage being used was taken by a SONY N50 digital video camera. The original resolution of the video was 720 pixels by 576 pixels (interlaced).

The video footage was deinterlaced and converted to a lower resolution of 352 by 288 pixels using “SUPER software” (Simplified Universal Player Encoder & Renderer) [91]. The converted video footage was compressed with a MPEG-2 algorithm with a bit rate of 8672 bps.

The total length of all sixteen videos was 35981 frames, which was approximately twenty four minutes of recording time at the frame rate of twenty five frames per second. A total number of 130 targets were manually identified in all sixteen videos.

A summary of the video footage used in this research is illustrated in table 7.2. In this table, the “Background” column identifies the complexity of the video footage, which was divided into three levels: simple, medium and complex, based on relative degree of background clutter, target size and appearance, camera stability, weather conditions and so on. The stabilities of the digital video camera during recording are listed in the “Stability” column. In this column, the word “Tripod” means that a tripod was used during video recording (stable) and the word “Handheld” means that the digital camera was held by hand (unstable) during video recording. The videos were taken at five different locations in Christchurch, New Zealand. They were: the University of Canterbury in Ilam,

Cathedral Square, Hareford Street, High Street and Cashel Street in the city centre. The location of every video is listed in the last column of table 7.2.

Video Name	Image Sample	Total Frames	Target Numbers	Background	Stability	Location
289.MPG		2717	7	Simple	Tripod	UOC
290.MPG		2774	13	Simple	Tripod	UOC
291.MPG		2702	9	Simple	Tripod	UOC
488.MPG		259	3	Complex	Handheld	Cathdral
489.MPG		9028	40	Complex	Handheld	Cathdral
490.MPG		1857	8	Complex	Handheld	Cathdral
491.MPG		2203	7	Complex	Handheld	Hareford
526.MPG		460	1	Medium	Handheld	Cashel
529.MPG		3628	11	Medium	Handheld	Cashel
530.MPG		792	3	Medium	Handheld	Cashel
531.MPG		317	1	Medium	Handheld	Cashel
532.MPG		532	3	Medium	Handheld	Cashel
533.MPG		2808	7	Medium	Handheld	Cashel
534.MPG		1181	3	Medium	Handheld	Cashel
535.MPG		1094	3	Medium	Handheld	Cashel
536.MPG		3629	11	Complex	Handheld	High

Table 7.2: A summary of the video footage used in this research.

7.3 The Experimental Method

The following steps were taken to pre-process the recorded video footage in section 7.2: Firstly, the original video footage was deinterlaced and converted to a lower 352 by 288 pixel resolution, which has been commonly used in prior computer vision based tracking research, such as [12]. Then all feasible targets were manually identified; their initial frame numbers, positions and sizes were recorded for tracker initialisation during experiments.

Next, these videos and target script files were fed into the human tracking system where the proposed tabu search particle filter and other metaheuristic particle filters were implemented. The position of the target in each frame and processing speed was recorded during experiments. Videos of the tracking process (the result video), which showed a tracker represented by a highlighted rectangular attached to a human target, were also generated during the experiment.

Finally, the experimental results were analyzed using three methods: 1) the recorded output videos were reviewed to count the number of targets that were incorrectly tracked. The number of incorrectly tracked targets shows the robustness of the tracking algorithm. The most robust tracking algorithm has the lowest incorrect tracking count. 2) The Euclidian distance between the ground truth and the tracking results were calculated. This distance represents the accuracy of a tracking algorithm. The most accurate tracking algorithm has the

smallest average distance between the ground truth and the tracking result. 3) The processing speeds of different metaheuristic particle filters were compared. The unit of process speed measurement used in this research is in seconds per frame.

7.4 The 'Ground Truth'

The 'ground truth' often refers to information that is collected "on location" in the area of remote image sensing technologies, such as aerial photographs or satellite imagery [92]. The concept of ground truth has been introduced into quantitative experimental problems and redefined as the 'accurate' or 'true' results of some measurements. In computer vision, manual selection, fiducial marker and synthesized footage are the three main methods to gather or generate ground truth for quantitative experiments. The shortcomings of these methods are reviewed in this section followed by an introduction of a unique method of calculating ground truth used in this research.

The first common method of specifying the ground truth of an object's position in a video footage is to manually specify the region of interest in each frame of the video. The shortcomings of this approach are: 1) the manual selection process is a very time consuming operation. The accuracy of the ground truth often decreases with time due to fatigue. That is, the manually specified ground truth is not consistent within a (long) video. 2) This method involves manual selection,

which is a very subjective process. Different people may have different opinions on the ground truth. For example, in the Berkeley segmentation dataset [93] for edge detection, humans disagreed with each other on the manually segmented ground truth images. Therefore, the F-score of the human segmentation is merely 0.79 [93]. 3) In the context of human tracking system, the target (a moving person) may change their body shape arbitrarily, making it difficult to consistently specify the locations of a human target.

The fiducial marker method is also often used for ground truth in vision based object tracking literature. In this approach, the ground truth of a system is given by tracking some reliable features, such as an ARToolKit marker [94, 95] or an infrared reflective ball [96]. Although the ground truth values specified through this method usually have a high standard of accuracy, it also has some shortcomings: 1) this method is intrusive. The target of a tracking system has to be modified so that the dedicated tracker can be installed. For example, a human target has to wear special clothes so that it can be tracked by an infrared tracking system. 2) The dedicated tracking systems are usually designed for very controlled environments. For example, infrared based tracking systems usually require that the intensity of ambient infrared radiation is minimized, and so this ground-truth acquisition technique cannot be used in bright sunlight which has a spectrum extending into the infrared region.

In contrast to the two methods introduced above, simulation based methods generate test video footage based on a known ground truth. One approach in producing simulated footage is to overlap a small image patch, for example the silhouette of a human body, onto a background video along some predefined trajectories (the known ground truth). Advanced techniques in earlier research include using a series of image patches iteratively to mimic the ego-motion of a human during walking or running, or alpha-blending the overlapped image patch with the background video so that the augmented image fits better into the video background. However, the most significant problem of such simulation based methods is that regardless of how many techniques are used to make the generated video footage look realistic, the mock-up footage is by no means comparable to a real one. Therefore, the test results based on simulated test footage is less reliable than those calculated from real footage.

In this research, a unique method of establishing ground truth information from arbitrary experiment video footage is developed based on the definition of Bayesian filters. The particle filter algorithm takes advantage of the Monte Carlo method to replace the intractable integration component in the theoretical Bayes filter by summing over discrete sets of Monte Carlo samples. As the number of Monte Carlo samples used in a particle filter increases towards infinity, the posterior belief distribution approximated by the particle filter approaches the posterior belief distribution calculated by the theoretical Bayes filter. Therefore, it

is possible to approach ground truth by allowing an extremely large number of particles to be used in the tracking system without any concern for computation resources and performance.

Under this definition, the aim of designing a 'good' tracking algorithm is redefined to approach the tracking results of the ground truth (calculated without any concern for computation resources and performance) within a reasonable timeframe using limited computation resources.

The actual process of approaching the ground truth with a very large number of particles was manually observed so that the tracker was always 'locked' onto the target. Figure 7.1 illustrates an intermediate stage of approaching the ground truth in a sample video. There were 100,000 particles used in this process and on average it took about 12.02 seconds to process one frame.

As shown in figure 7.1, the particles are colour coded based on their weights. Particles with cold colours, such as blue, are the ones with low weights; particles with warm colours, such as red, are the ones with high weights. White coloured particles shown in the middle of red mark the most important particles and so are the 'best' particle found in the current iteration (the maximum a posterior approach).

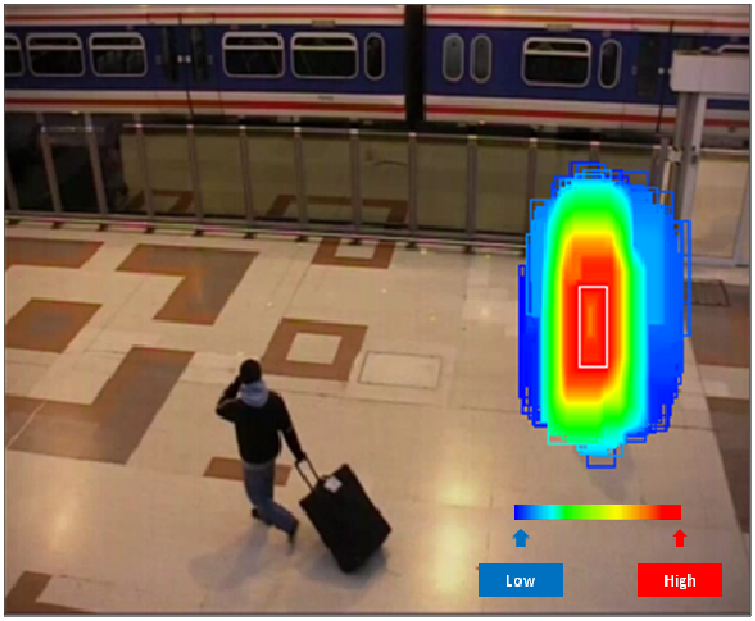


Figure 7.1: An image shows that the 'ground truth' is calculating in process with 100,000 particles.

7.5 Experiment Result and Analysis

In this section, the experimental data gathered from the human tracking system is illustrated and analysed.

Three aspects for the proposed algorithms were evaluated. Firstly the robustness of the algorithm was indicated by the number of missed targets. A smaller number of missed targets indicate that the corresponding tracking algorithm was more robust. The missed targets were manually counted by reviewing the output videos of the tracking process. For a given target, if the tracker's region of interest was completely detached from the subject at any given time (regardless

the number of frames missing), this target was counted as a missed target. Two examples of missed targets are illustrated in figure 7.2.

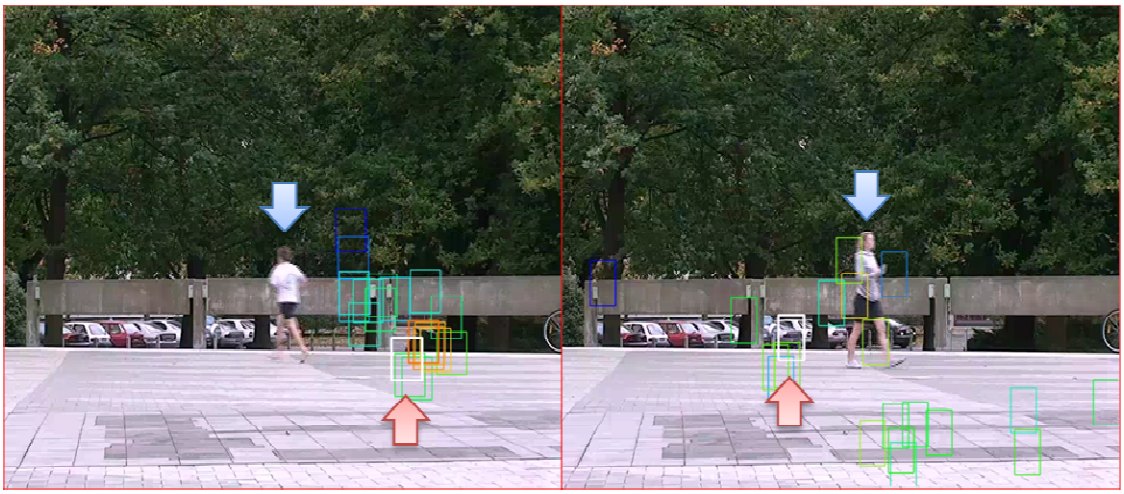


Figure 7.2: Two examples of the missing targets during tracking process. As shown in both images, the blue coloured arrows point to the correct position of the target whereas the red coloured arrows point to the incorrect position indicated by the tracker (white rectangle).

The second aspect is the accuracy of the algorithm. The accuracy is indicated by the average tracking error, that is, the Euclidian distance between the ground truth positions and the positions calculated during experiments. A more accurate tracking algorithm is indicated by a smaller tracking error. The tracking error was calculated automatically by averaging the difference between the ground truth and the experiment output. The corresponding target in both files was identified by the video footage name, target name and frame number during this automatic process. Figure 7.3 illustrates two examples of the tracking error between the ground truth (red colour) and experiment outputs (green colour). The experiment

output illustrated in the figure on the right is more accurate than the one shown on the left.

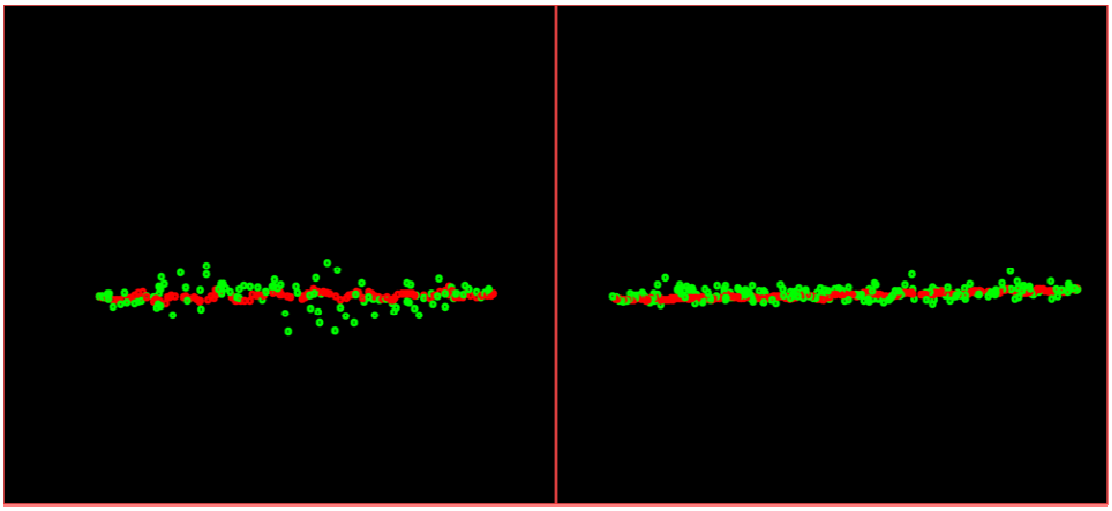


Figure 7.3: Two examples of the tracking accuracy evaluation. The red coloured dots indicate the positions of the ground truth whereas the green dots indicate the positions calculated by a tracker.

The final aspect of the proposed algorithms evaluated in this research is the speed of processing, which is indicated by the average elapsed processing time per frame. A fast algorithm has a low elapsed time per frame value while a slow algorithm has a large value. The elapsed time used in this research was calculated by the Windows' "QueryPerformanceCounter()" API function, which is a precise time base [97].

7.5.1 The Robustness of the Tabu Search Filter and Other Metaheuristic particle filters

The first experiment conducted in this research was to evaluate the robustness of the proposed tabu search particle filter and compare it to other existing metaheuristic particle filters, including the evolutionary filter (EV) [55], the simulated annealing (SA) filter [49], the particle swarm (PS) filter [57], and the scatter search (SS) and path relink (PR) filters [54].

As discussed earlier, the robustness of a tracking algorithm is inversely proportional to the number of targets that are incorrectly tracked (missing) in the experiment. The missing targets for each algorithm evaluated in this research are listed in table 7.3. These numbers were collected by manually reviewing the 960 video clips (six algorithms by sixteen video footage by ten test conditions) that record the tracking results.

PNUM	TS	EV	SA	PS	SS	PR
10	8	75	36	107	55	41
20	6	64	37	107	44	32
30	6	67	31	119	45	34
40	8	64	28	103	51	34
50	8	52	26	104	49	36
60	7	49	25	97	49	37
70	8	45	23	94	50	39
80	9	46	22	93	46	31
90	9	45	27	92	46	40
100	8	40	21	89	41	36

Table 7.3: The number of missing targets of the six metaheuristic particle filters evaluated in the research.

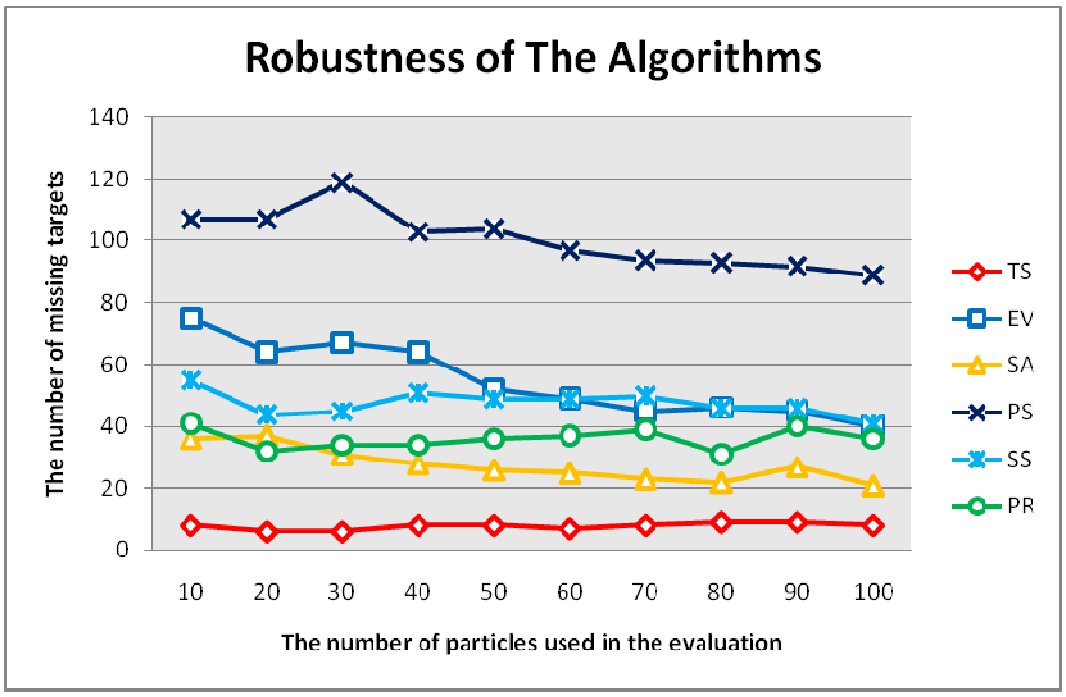


Figure 7.4: The robustness results of the six tracking algorithms, as evaluated with ten different particle numbers.

Robustness of the evaluated algorithms is graphed in figure 7.4 based on the missing target count listed in table 7.3. As shown, the tabu search particle filter has the lowest missing target count, which varies from six targets to nine targets. On the other hand, the particle swarm filter performs worst among the six algorithms evaluated. On average, the number of missed targets for the particle swarm filter is 100.5 targets, which is 13.05 times worse than the tabu search particle filter.

Furthermore, as clearly shown in table 7.3 and figure 7.4, the robustness of the proposed algorithm does not rely on a large number of particles. It achieves

93.85% of the correct tracking rate with merely 10 particles whereas under the same condition, the second best algorithm, the simulated annealing filter, has a success rate of only 72.31%.

7.5.2 The Accuracy of the Tabu Search Filter and Other Metaheuristic particle filters

The accuracy of the tracking algorithms are evaluated and compared in this section. As mentioned previously, the accuracy of an algorithm is measured by the average Euclidian distances between the ground truth and the actual tracking results over all targets. Table 7.4 lists the average tracking errors of all implemented algorithms. As shown, ten configurations were used during this test and the particle numbers are increased from ten to one hundred with a step size of ten particles. The accuracy results listed in the table 7.4 are shown graphed in figure 7.5.

PNUM	TS	EV	SA	PS	SS	PR
10	2.69	70.21	10.34	110.84	36.83	15.52
20	1.89	27.80	2.88	105.94	16.61	11.20
30	1.50	23.76	1.76	90.28	15.38	10.65
40	1.37	15.92	1.51	88.95	14.37	10.05
50	1.69	8.83	1.48	79.57	18.26	9.98
60	1.24	11.22	1.53	91.68	13.46	10.71
70	1.18	9.27	1.27	71.05	13.79	10.20
80	1.10	8.14	1.19	60.12	12.69	8.45
90	1.49	7.85	1.19	74.52	12.47	8.31
100	1.06	11.66	1.23	69.40	12.10	8.24

Table 7.4: The average Euclidian distances between the ground truth positions and the positions calculated during experiment.

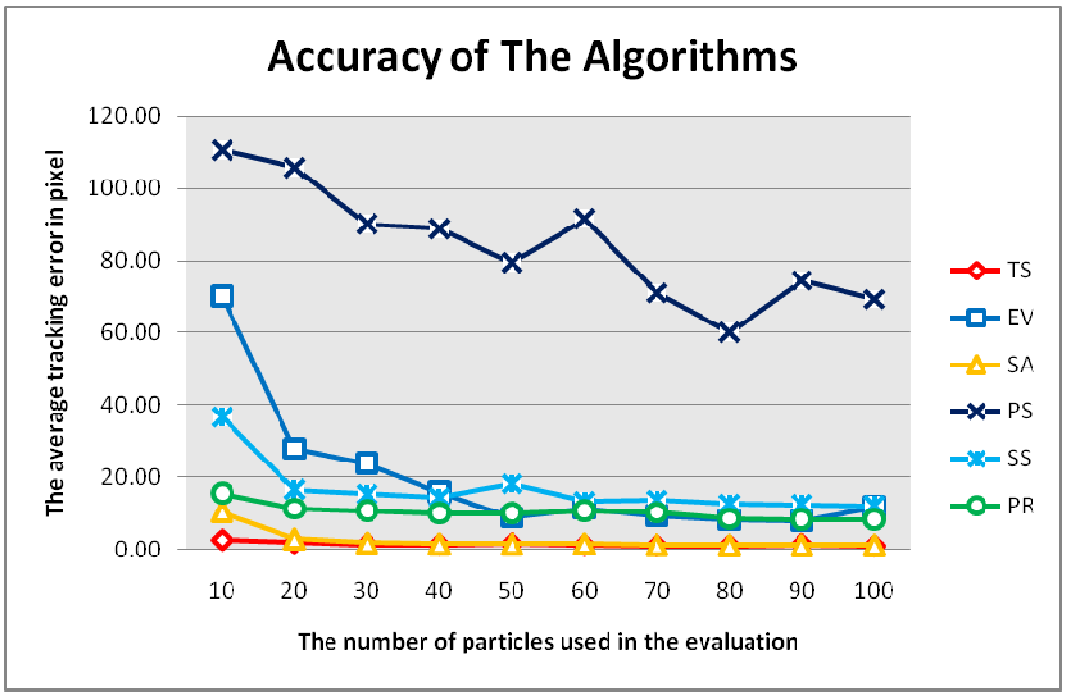


Figure 7.5: The accuracy results of the six tracking algorithms, as evaluated with ten different particle numbers.

As shown in figure 7.5, the tracking accuracy results of all algorithms evaluated in this research demonstrate a trend that the tracking error decreases when the population of particles is increased. The tabu search filter also follows this trend and it achieved an average tracking error of 2.69 pixels when only ten particles were used. The average tracking accuracy dropped to less than half, with a value of only 1.06 pixels when the particles were increased to a hundred.

Figure 7.5 shows that the simulated annealing algorithm approaches the tracking accuracy of the tabu search particle filter only when a large number of particles are used. But when only ten particles were used, the tabu search particle filter

was 3.84 times more accurate than the second best algorithm, being the simulated annealing filter.

Figure 7.6 takes a closer look at the accuracy results of the tabu search filter and the simulated annealing filter. As shown, although both algorithms had similar tracking accuracy, when there were more than 30 particles used in the experiment, the tabu search filter clearly achieves much better accuracy than the simulated annealing algorithm when the particle numbers are small. Furthermore, it is clear from figure 7.5 that the accuracy of the tabu search particle filter was also significantly improved when the particle numbers were increased. By using 100 particles, the new tabu search particle filter was about 2.54 times more accurate than using 10 particles.

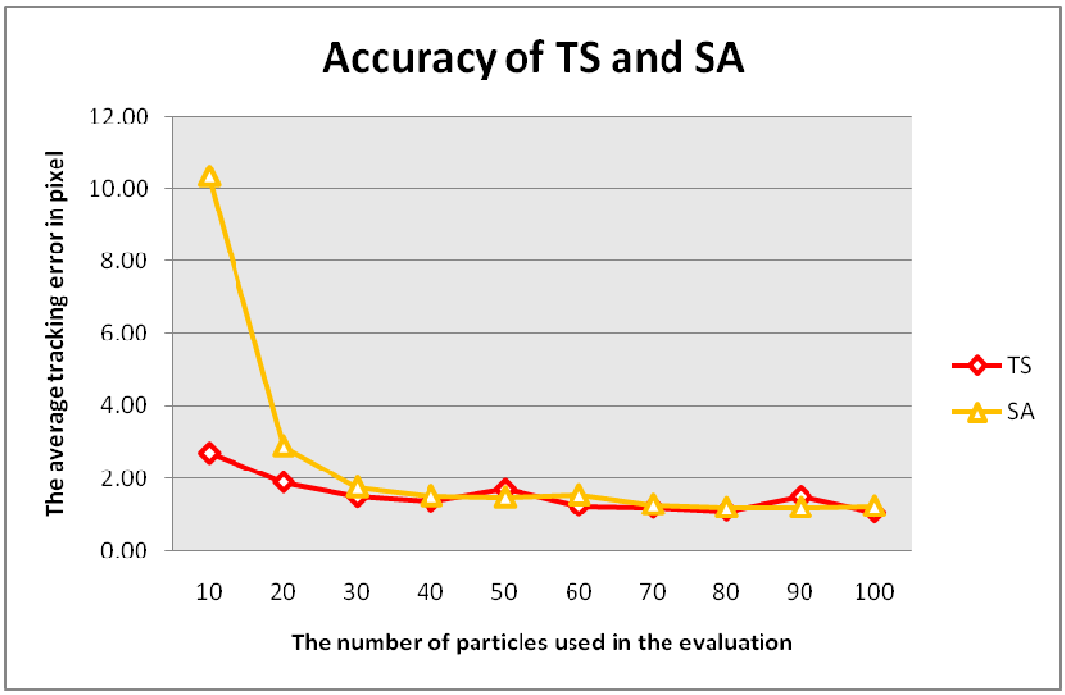


Figure 7.6: A clearer comparison of the accuracy results of only the tabu search and simulated annealing filters, as evaluated with ten different particle numbers.

7.5.3 The Speed of the Tabu Search Filter and Other Metaheuristic particle filters

In this section the speed of the tabu search particle filter is evaluated and compared with other metaheuristic particle filters. The speed of a tracking algorithm is measured from the average number of seconds used to process a frame in a given footage.

The average speed for all six algorithms is listed in table 7.5 and graphed in figure 7.7. As shown, the proposed algorithm is faster than all other algorithms

evaluated in this research at all times. For example, when only ten particles were used, the tabu search particle filter was approximately six times faster than the simulated annealing algorithm. When a hundred particles were used, the speed difference between these two algorithms was increased from approximately 6 times to approximately 7.6 times. Furthermore, as illustrated in table 7.5, the proposed algorithm was 60 times faster than the slowest algorithm evaluated in the experiment when only 10 particles are used.

PNUM	TS	EV	SA	PS	SS	PR
10	0.02	0.03	0.12	0.13	1.20	0.42
20	0.04	0.06	0.25	0.19	1.32	0.60
30	0.06	0.09	0.37	0.26	1.73	0.79
40	0.07	0.12	0.49	0.32	2.00	1.06
50	0.09	0.16	0.63	0.41	2.46	1.31
60	0.11	0.18	0.75	0.45	2.66	1.50
70	0.12	0.20	0.85	0.54	3.16	1.75
80	0.13	0.23	0.97	0.61	3.47	1.90
90	0.15	0.25	1.08	0.60	3.58	2.22
100	0.16	0.28	1.22	0.68	3.94	2.48

Table 7.5: The average processing speed calculated during experiment.

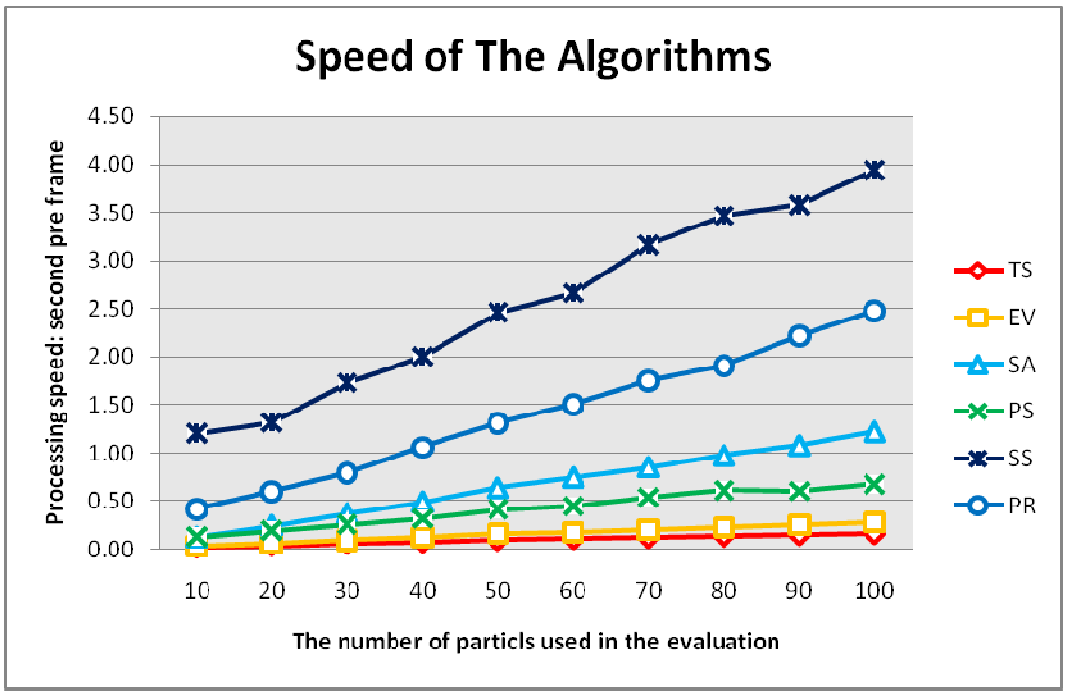


Figure 7.7: The speed results of the six tracking algorithms evaluated with ten different particle numbers.

7.6 Summary

In this chapter, the proposed tabu search particle filter was evaluated quantitatively in unconstrained human tracking environments. The experimental results show that the tabu search particle filter outperforms all other metaheuristic particle filters in all three aspects, namely robustness, accuracy and speed.

Chapter 8
Conclusion and Future Work

The motivation for this research was to achieve real time tracking in unconstrained environments. The experimental results demonstrate that the new tabu search particle filter outperforms other metaheuristic particle filters (including evolution particle filter, particle swarm filter, annealing particle filter, path relink filter and scatter search filter). The proposed algorithm, based on an integration of the genetic particle filter and modified tabu search metaheuristic, outperformed these other metaheuristic particle filters in all aspects of robustness, accuracy and speed.

Even with only ten particles in the system, the proposed tabu search particle filter achieved 93.85% successful tracking whereas the success rate of other metaheuristic particle filters ranged from 68.46% to 17.69% under the same tracking condition. The accuracy of the new algorithm (using only ten particles) averaged 2.69 pixels, which was 3.85 times better than the second best metaheuristic particle filter and 18.13 times better than the average accuracy of all other filters. The proposed algorithm was also the fastest among all metaheuristic particle filters that have been tested. It achieves approximately 50 frames per second, which is 1.5 times faster than the second fastest algorithm (the evolution particle filter) and nineteen times faster than the average speed of all other metaheuristic particle filters.

In addition, a colour sequence model was developed in this research based on a degenerated form of the hidden Markov model. The motivation of developing this unique model is to have an illumination invariant observation model. Quantitative evaluation based on rigid object matching experiments illustrate that the successful matching rate of the colour sequence model is 5.73 times better than the widely used colour histogram. In terms of speed, the new algorithm achieves 2.09 times better successful matching rate in about three quarters of the processing time consumed by the colour histogram model.

The quantitative experiments demonstrated that high quality tracking results could be achieved using the new tabu search particle filter, even when the number of particles used in the system was very small. This can be a great advantage when the proposed algorithm is tracking huge numbers of targets or on the platforms with limited resources, such as microcontrollers. Research towards building a robust and accurate real time tracking system on embedded platforms such as cell phones will be conducted in the future based on the theory proposed in this thesis.

Furthermore, in the new colour sequence model, the transition probability between adjacent hidden states is degenerated to binary values. That is, the transition probability between two states can only be either one, which indicates a certain transition, or zero, which indicates an invalid transition. In this case, one colour sequence model can be

used to represent only one coloured object. However, by using a more sophisticated hidden Markov model structure and continuous transition probabilities among hidden states, the proposed colour sequence model can be extended to represent multiple coloured objects in one model. Future research can be conducted in this direction as well to extend the capability of the proposed colour sequence model.

References

[1] M. Isard, and A. Blake, “CONDENSATION, Conditional Density Propagation for Visual Tracking ” *International Journal of Computer Vision,* vol. 29, no. 1, pp. 5 - 28, 1998.

[2] C. S. Fahn, and Y. T. Lin, “Real-time Face Tracking Techniques Used for the Interaction between Humans and Robots,” in IEEE Conference on Industrial Electronics and Applications, Beijing, China, 2010, pp. 12 - 17.

[3] M. Nieto, C. Cuevas, and L. Salgado, “Measurement-based Reclustering for Multiple Object Tracking with Particle FIlters,” in IEEE International Conference on Image Processing, Cairo, Egypt, 2010, pp. 97 - 100.

[4] F. Mustiere, M. Bolic, and M. Bouchard, “Rao-Blackwellised Particle Filters: Example of Applications,” in Canadian Conference on Electrical and Computer Engineering, Ottawa, Canada, 2006, pp. 196 - 200.

[5] J. Madapura, and B. X. Li, “3D Articulated Human Body Tracking using KLD-Annealed Rao-Blackwellised Particle Filter,” in International Conference on Multimedia and Expo, Beijing, China, 2007, pp. 950 - 953.

[6] H. Loose, U. Franke, and C. Stiller, “Kalman Particle Filter for Lane Recognition on Rural Roads,” in IEEE Symposium on Intelligent Vehicles, Xi'an, China, 2009, pp. 60 - 65.

[7] L. Mao, and S. Liu, “Multisensor Information Fusion Extended Kalman Particle Filter,” in International Conference on Advanced Computer Control, Shenyang, China, pp. 417 - 419.

[8] L. Li, H. Ji, and J. Luo, “The Iterated Extended Kalman Particle Filter,” in IEEE International Symposium on Communications and Informantion Technology, Beijing, China, pp. 1213 - 1216.

[9] X. Deng, and P. Zhou, “A New Interacting Multiple Model Algorithm Based on the Unscented Particle Filter,” in International Conference on Information Assurance and Security, Xi'an, China, pp. 419 - 422.

[10] A. Wang, J. Li, and A. Yan, “The Semi-Iterative Unscented Particle Filtering,” in International Workshop on Intelligent System and Applications, Wuhan, China, pp. 1 - 4.

[11] Y. Shi, and C. Han, “The Divided Difference Particle Filter,” in International Conference on Information Fusion, Quebec, Canada, pp. 1 - 7.

[12] S. Schreiber, and G. Rigoll, “Omni-directional Multiperson Tracking in Meeting Scenarios Combining Simulated Annealing and Particle Filtering,” in IEEE International Conference on Automatic Face and Gesture Recognition, Amsterdam, pp. 1 - 6.

[13] X. Liang, J. Feng, Q. Li *et al.*, “A Swarm Intelligence Optimization for Particle Filter,” in International Conference on Intelligent Control and Automation, Chongqing, China, pp. 1986 - 1991.

[14] L. Ye, J. Wang, C. Li *et al.*, “A Novel Particle Filtering Framework Using Genetic Monte Carlo Sampling,” in International Conference on Management and Service Science, Wuhan, China, pp. 1 - 4.

References

[15] M.-C. Ho, C.-C. Chiang, and Y.-L. Chen, "A Genetic Particle Filter for Moving Object Tracking," in International Conference on Image and Graphics, Si Chuan, China, 2007, pp. 524 - 529.

[16] J. J. Pantrigo, and A. Sanchez, "Hybridizing Particle FIlters and Population-based Metaheuristics for Dynamic Optimization Problems," in International Conference on Hybird Intelligent Systems., Rio de Janeiro, Brazil, pp. 68 - 83.

[17] S. Feng, Q. Guan, S. Xu *et al.*, "Human Tracking Based on Mean Shift and Kalman Filter," in International Conference on Artificial Intelligence and Computational Intelligence, Shanghai, China, pp. 518 - 522.

[18] M. Lee, R. Green, and M. Billinghurst, "3D Natural Hand Interaction for AR Applications," in International Conference Image and Vision Computing New Zealand, Christchurch, New Zealand, 2008.

[19] A. Clark, "OPIRA: The Optical-flow Perspective Invariant Registration Augmentation and Other Improvements for Natural Feature Registration," PhD Thesis, Computer Science and Software Engineering, University of Canterbury, Christchurch, 2009.

[20] I. Dotu, M. A. Patricio, A. Berlanga *et al.*, "Boosting video tracking performance by means of Tabu Search in intelligent visual surveillance systems," *Journal of Heuristics,* vol. 16, no. 5, pp. 633 - 659, 2010.

[21] Y. Bar-Shalom, F. Daum, and J. Huang, "The Probabilistic Data Association Filter," *IEEE Control System Magazine,* vol. 10, pp. 82 - 100, 2009.

[22] F. Daum, "Multitarget-Multisensor Tracking: Principles and Techniques," *IEEE Aerospace and Electronic Systems Magazine,* vol. 11, no. 2, pp. 41 - 51, 1996.

[23] S. S. Blackman, "Multiple Hypothesis Tracking for Multiple Target Tracking," *IEEE Aerospace and Electronic Systems Magazine,* vol. 19, no. 1, pp. 5 - 18, January, 2004, 2004.

[24] A. G. Daronkolaei, V. Nazari, M. B. Menhaj *et al.*, "A Joint Probability Data Association Filter Algorithm for Multiple Robot Tracking Problems," *Tools in Artificial Intelligence*, P. Fritzsche, ed., Austria: InTech, 2008.

[25] D. Duque, H. Santos, and P. Cortez, "Moving Object Detection Unaffected by Cast Shadows, Hightlight and Ghosts," in International Conference on Image Processing, Cenova, Italy, 2005, pp. 413 - 416.

[26] D. Wedge, D. Huynh, and P. Koves, "Tracking Footballs through Clutter in Broadcast Digital Videos," in International Conference on Image and Vision Computing, Akaroa, New Zealand, 2004, pp. 155- 160.

[27] C. Fang, *Gaussian Mixture Model Background Modeling for Video*, University of Cincinnati, Cincinnati, USA, 2008.

[28] K. Kim, T. H. Chalidabhongse, D. Harwood *et al.*, "Real-time Foreground-Background Segmentation using Codebook Model," *Journal of Real-time Imaging,* vol. 11, no. 3, pp. 167 - 256, June, 2005, 2005.

[29] Proc. Int. Joint Conf. on Computational Sciences and Optimization, Sanya, Hainan, China, April 24–26, 2009, vol. 2, pp. 273 – 277, 2009.

[30] K. Kim, T. H. Chalidabhongse, D. Harwood *et al.*, "Background Modeling and Subtraction by Codebook Construction," in IEEE International Conference on Image Processing, Singapore, 2004, pp. 233 - 242.

References

[31] J. Ma, C. Han, and Y. Chen, “Efficient Visual Tracking Using Particle Filter,” in Australasian Research Management Society Annual Conference, Australia, 2007, pp. 17 - 23.

[32] J. Zhu, Y. Lao, and Y. Zheng, “Object Tracking in Structured Environments for Video Surveillance Applications,” *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 20, no. 2, pp. 223 - 236, 2010.

[33] R. Kalman, “A New Approach to Linear Filtering and Prediction Problems,” *Transactions of the ASME--Journal of Basic Engineering,* vol. 82, no. D, pp. 35 - 45, 1960.

[34] P. Swerling, *A Proposed Stagewise Differential Correction Procedure for Satellite Tracking and Prediction*, RAND Corporation, Santa Monica, USA, 1958.

[35] S.Y. Lin, I.-C. Chang, Dynamic Kernel-Based Progressive Particle Filter for 3D Human Motion Tracking, in: Proc.Asian Conf. on Computer Vision (ACCV 2009), Xi’an, China, Sept. 23 – 27, 2009, vol. 2 (Lecture Notes in Computer Science, vol. 5995), Springer-Verlag: Berlin, pp.257-266, 2009.

[36] B. Ristic, S. Arulampalam, and N. Gordon, "Suboptimal Nonlinear Filters," *Beyond The Kalman Filter: Particle Filters for Tracking Applications*, Boston: Artech House, 2004.

[37] S. Thrun, W. Burgard, and D. Fox, "Gaussian Filters," *Probabilistic Robotics*, pp. 39 - 84, Cambridge: The MIT Press, 2005.

[38] Y. T. Chan, A. G. C. Hu, and J. B. Plant, “A Kalman Filter Based Tracking Scheme with Input Estimation,” *IEEE Transactions on Aerospace and Electronic Systems,* vol. 15, no. 2, pp. 237 - 244, 1979.

[39] B. D. R. Stenger, P. R. S. Mendonca, and R. Cipolla, “Model-based Hand Tracking Using an Unscented Kalman Filter,” in British Machine Vision Conference, Manchester, UK, 2001, pp. 401 - 410.

[40] M. St-Pierre, and D. Gingras, “Comparison Between the Unscented Kalman Filter and the Extended Kalman Filter for the Position Estimation Module of an Integrated Navigation Information System,” in International Symposium on Intelligent Vehicles, Las Vegas, USA, 2004, pp. 831 - 835.

[41] S. J. Julier, and J. K. Uhlmann, “A New Extension of the Kalman Filter to Nonlinear Systems,” in International Conference on Aerospace Defence Sensing, Simulation and Controls, Orlando, USA, 1997.

[42] "Extended Kalman filter From Wikipedia, the free encyclopedia," March, 2009; http://en.wikipedia.org/wiki/Extended_Kalman_filter.

[43] M. Bolic, P. M. Djuric, and S. Hong, “Resampling Algorithms and Architectures for Distributed Particle Filters,” *IEEE Transactions on Signal Processing,* vol. 53, no. 7, pp. 2442- 2450, 2005.

[44] M. Bolić, P. M. Djurić, and S. Hong, “Resampling Algorithms for Particle Filters: A Computational Complexity Perspective,” *Journal on Applied Signal Processing,* vol. 15, pp. 2267 - 2277, 2004.

[45] M. Bolić, P. M. Djurić, and S. Hong, *Resampling Algorithms and Architectures for Distributed Particle Filters*, Stony Brook University, New York, USA, 2004.

References

[46] A. Athalye, M. Bolic, S. Hong *et al.*, "Generic Hardware Architectures for Sampling and Resampling in Particle Filters," *Journal of Applied Signal Processing,* vol. 17, pp. 2888-2902, 2005.

[47] M. B. S. Hong, and P. M. Djuric, "An Efficient Fixed-Point Implementation of Residual Systematic Resampling Scheme for High-Speed Particle Filters," *IEEE Signal Processing Letters,* vol. 11, no. 5, 2004.

[48] J. D. Hol, T. B. Schon, F. Gustafsson, "On Resampling Algorithms for Particle Filters," in Nonlinear Statistical Signal Processing Workshop, Cambridge, UK, 2006.

[49] J. Deutscher, A. Blake, and I. Reid, "Articulated Body Motion Capture by Annealed Particle Filtering," in Conference on Computer Vision and Pattern Recognition Hilton Head, USA, 2000, pp. 126 - 133.

[50] J. Deutscher, and I. Reid, "Articulated Body Motion Capture by Stochastic Search," *International Journal of Computer Vision,* vol. 61, no. 2, pp. 185 - 205, 2005.

[51] M. Fontmarty, F. Lerasle, and P. Danes, "Data Fusion within a Modified Annealed Particle Filter Dedicated to Human Motion Capture," in International Conference on Intelligent Robots and Systems, 2007, pp. 391 - 396.

[52] J. Darby, B. Li, and N. Costen, "Tracking a Walking Person using Activity-Guided Annealed Particle Filtering," in International Conference on Automatic Face and Gesture Recognition, Amsterdam, 2008, pp. 1 - 6.

[53] C. Canton-Ferrer, J. R. Casas, and M. Pardas, "Voxel based Annealed Particle Filtering for Markerless 3D Articulated Motion Capture," in International Conference on 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, Potsdam, 2009, pp. 1 - 4.

[54] J. J. Pantrigo, A. Sanchez, K. Gianikellis *et al.*, "Combining Particle Filter and Population-based Metaheuristics for Visual Articulated Motion Tracking," *Electronic Letters on Computer Vision and Image Analysis,* vol. 5, no. 3, pp. 68 - 83, 2005, 2005.

[55] N. M. Kwok, G. Fang, and W. Zhou, *Evolutionary Particle Filter: Resampling from the Genetic Algorithm Perspective*, University of Western Sydney, Penrith, 2005.

[56] S. Park, J. Hwang, K. Rou *et al.*, "A New Particle Filter Inspired by Biological Evolution: Genetic Filter," *World Academy of Science, Engineering and Technology,* vol. 33, 2007, 2007.

[57] C. J. Ji, Y. Y. Zhang, M. M. Tong *et al.*, "Particle Filter with Swarm Move for Optimization," in International Conference on Parallel Problem Solving from Nature, Dortmund, Germany, 2008, pp. 909 - 918.

[58] Y. H. Zheng, and Y. Meng, "Swarming Particles with Multi-feature Model for Free-Selected Object Tracking," in International Conference on Intelligent Robots and Systems, Nice, France, 2008, pp. 553 - 558.

[59] X. Q. Zhang, W. M. Hu, and S. Maybank, "A Smarter Particle FIlter," in Asian Conference on Computer Vision, Xi An, China, 2009, pp. 100 - 111.

[60] A. J. Monticelli, R. Romero, and E. N. Asada, "Fundamentals of Simulated Annealing," *Modern Heuristic Optimization Techniques*, K. Y. Lee and M. A. El-Sharkawi, eds., pp. 123 - 144, Piscataway, USA: IEEE Press, 2008.

[61] S. Schreiber, and G. Rigoll, “Omni-directional Multiperson Tracking in Meeting Scenarios Combining Simulated Annealing and Particle Filtering,” in International Conference on Automatic Face and Gesture Recognition, 2008, pp. 1 - 6.

[62] H. Q. Yue, C. R. Li, Y. X. Liang *et al.*, “Tracking Human Arm from Monocular Videos,” in Internation Conference on Mechatronics and Automation, 2007, pp. 155 - 159.

[63] J. Li, H. Yu, L. Zhou *et al.*, “UPF Tracking Algorithm Based on Color Distribution and Simulated Annealing,” in International Conference on Mechatronics and Automation, Harbin, China, 2007, pp. 429 - 434.

[64] J. Li, H. Yu, L. Zhou *et al.*, “An Adaptive Unscented Particle Filter Tracking Algorithm Based on Color Distribution and Wavelet Moment,” in Industrial Electronics and Applications, Harbin, China, 2007, pp. 218 - 223.

[65] J. Dreo, A. Petrowski, P. Siarry *et al.*, "Tabu Search," *Metaheuristics for Hard Optimization*, pp. 47 - 72, Berlin, Germany: Springer-Verlag., 2006.

[66] M. Gendreau, and J.-Y. Potvin, "Tabu Search," *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, E. K. Burke and G. Kendall, eds., pp. 165 - 186, New York, USA: Springer Science + Business Media, Inc., 2005.

[67] R. Grant, “Signal-linear Representations of Colour for Computer Vision,” PhD Thesis, Computer Science and Software Engineering, University of Canterbury, Christchurch, 2010.

[68] R. D. Green and L. Guan, “Quantifying and recognizing human movement patterns from monocular video Images-part I: a new framework for modeling human motion,” *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 14, no. 2, pp. 179 - 190, February, 2004, 2004.

[69] R. D. Green and L. Guan, “Quantifying and recognizing human movement patterns from monocular video images-part II: applications to biometrics,” *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 14, no. 2, pp. 191 - 198, Feburary, 2004, 2004.

[70] K. Koyamada, S. Tamura, and O. Ono, "Color Based Tracking Vision for the Intelligent Space," *Systems Modeling and Simulation*, pp. 420 - 424: Springer Japan, 2007.

[71] M. Kristan, J. Perš, M. Perše *et al.*, “Closed-World Tracking of Multiple Interacting Targets for Indoor-Sports Applications,” *Computer Vision and Image Understanding,* vol. 113, no. 5, pp. 598-611, 2009.

[72] F. Li, and R. J. Woodham, “Video Analysis of Hockey Play in Selected Game Situations,” *Image and Vision Computing,* vol. 27, no. 2, pp. 45-58, 2009.

[73] G. Liu, X. Tang, H. D. Cheng *et al.*, “A Novel Approach for Tracking High Speed Skaters in Sports Using a Panning Camera,” *Pattern Recognition,* vol. Accepted Manuscript, 2009.

[74] J. Liu, X. Tong, W. Li *et al.*, “Automatic Player Detection, Labeling and Tracking in Broadcast Soccer Video,” *Pattern Recognition Letters,* vol. 30, no. 2, pp. 103-113, 2009.

[75] W.-L. Lu, K. Okuma, and J. J. Little, “Tracking and Recognizing Actions of Multiple Hockey Players using the Boosted Particle Filter,” *Image and Vision Computing,* vol. 27, no. 2, pp. 189-205, 2009.

References

[76] M. Perše, M. Kristan, S. Kovačič *et al.*, "A Trajectory-based Analysis of Coordinated Team Activity in a Basketball Game," *Computer Vision and Image Understanding,* vol. 113, no. 5, pp. 612-621, 2009.

[77] M. A. Sotelo, F. J. Rodriguez, L. Magdalena *et al.*, "A Color Vision-Based Lane Tracking System for Autonomous Driving on Unmarked Roads," *Journal of Autonomous Robots,* vol. 16, no. 1, pp. 95 - 116, 2004.

[78] X. Zhang, and J. Yang, "The Analysis of the Color Similarity Problem in Moving Object Detection," *Signal Processing,* vol. 89, no. 4, pp. 685-691, 2009.

[79] H. Zhou, Y. Yuan, Y. Zhang *et al.*, "Non-rigid Object Tracking in Complex Scenes," *Pattern Recognition Letters,* vol. 30, no. 2, pp. 98 - 102, 2009.

[80] P. Blunsom. "Hidden Markov Models," January, 2009. Accessed in August, 2011.

[81] Z. Ghahramani, "An Introduction to Hidden Markov Models and Bayesian Networks," *Hidden Markov Models Applications in Computer Vision*, H. Bunke and T. Caelli, eds., pp. 9 - 42, Singapore: World Scientific Publishing Co. Pte. Ltd., 2001.

[82] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE,* vol. 77, no. 2, pp. 257 - 286, February, 1989, 1989.

[83] R. Schwartz, *Biological Modeling and Simulation A Survey of Practical Models, Algorithms, and Numerical Methods*, pp. 291 - 308, Cambridge, MA: Massachusetts Institute of Technology Press, 2008.

[84] O. Cappe, E. Moulines, and T. Ryden, *Inference in Hidden Markov Models*, New York: Springer Science Business Media, Inc., 2005.

[85] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory,* vol. 13, no. 2, pp. 260-269, April, 1967.

[86] D. McGraw, and J. Wagner, "Elliptically symmetric distributions," vol. 14, no. 1, pp. 110- 120, June, 1968.

[87] S. Kullback, and R. A. Leibler, "On Information and Sufficiency," *The Annals of Mathematical Statistics,* vol. 22, no. 1, pp. 79 - 86, March, 1951.

[88] Y. B. Lee, and J. R. Deller, "Heuristic Structural Modifications to the HMM for Efficient Resource Utilization," in IEEE International Conference on Acoustics, Speech, and Signal Processing, New York, USA, 2003, pp. 273 - 276.

[89] "Logarithmically concave function - From Wikipedia, the free encyclopedia," February, 2009; http://en.wikipedia.org/wiki/Logarithmically_concave_function.

[90] T. Bailey, *Summary of Probability Rules*, University of Sydney, Sydney, 2006.

[91] "Simplified Universal Player Encoder & Renderer," ERightSoft, 2010.

[92] "Ground truth - From Wikipedia, the free encyclopedia," April, 2009; http://en.wikipedia.org/wiki/Ground_truth.

[93] R. Hidayat, "Real-time Texture Boundary Detection," PhD Thesis, Computer Science and Software Engineering, University of Canterbury, Christchurch, New Zealand, 2010.

[94] E. Woods, P. Mason, and M. Billinghurst, "MagicMouse: an Inexpensive 6-Degree-of-Freedom Mouse," in Graphite, Melbourne, Australia, 2003.

[95] R. Grasset, A. Duenser, and M. Billinghurst, “The Design of a Mixed-Reality Book: Is It Still a Real Book?,” in International Symposium on Mixed and Augmented Reality, Cambridge, UK, 2008.

[96] H. Z. Gao, M. Lim, E. Lin *et al.*, “Marker based Facial Tracking Application in Communication Disorder Research,” in Image and Vision Computing New Zealand, Auckand, New Zealand, pp. 27 - 33.

[97] "Beware of QueryPerformanceCounter()," 2010; http://www.virtualdub.org/blog/pivot/entry.php?id=106.

[98] Y. Zhang, N. Li, Y. Zhao, 3-D Human Tracking Based on Particle Filter, in: Proc. Int. Joint Conf. on Computational Sciences and Optimization, Sanya, Hainan, China, April 24–26, 2009, vol. 2, pp. 273 – 277, 2009.

[99] D. M. Hansen, P. T. Dulzer, S. Park e.a., Multi-view Video Analysis of Human and Vehicles in an Unconstrained Environment, in: Proc. 4th Int. Symp. on Advances in Visual Computing, Las Vegas, NV, USA, Dec. 1 – 3, 2008 (Lecture Notes in Computer Science, vol. 5358), Springer- Verlag: Berlin, pp. 428 – 439, 2008.

[100] C. Sminchisescu, B. Triggs, Estimating Articulated Human Motion with Covariance Scaled Sampling, Int. Journal of Robotics Research, vol. 22, no. 6, pp. 371 – 391, 2003.

[101] A. Yilmaz, O. Javed, M. Shah, Object tracking: A survey, ACM Computing Surveys (CSUR), v.38 n.4, p.13-es, 2006.

[102] C. L. Devasena, R. Revathi, M. Hemalatha: Video Surveillance Systems - A Survey. International Journal of Computer Science (IJCSI)., Vol. 8, Issue 4, No 1. July, 2011.